DO THE SAME HINTS MATTER ACROSS INPUT SIZES? INTERVENING ON SEMANTIC INTERMEDIATE STATES IN NEURAL ALGORITHMIC REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural algorithmic reasoners often generalize poorly to inputs larger than those seen during training. Although CLRS models are supervised with intermediate algorithmic states (HINTs), it remains unclear whether these states retain the same functional role when input size increases. Unlike opaque latent activations, CLRS HINTs correspond to named algorithmic variables with exact ground-truth trajectories, allowing their role in recurrent execution to be tested directly through intervention.

We study this question across 28 CLRS algorithms and two recurrent processors at $N = 16$ and $N = 64$. We separately test the effect of restoring an individual HINT to its ground-truth trajectory, disrupting its input-specific semantic value through valid semantic resampling, and changing the surrounding HINT context in which the same HINT acts. Both restoration and semantic dependence retain cross-size structure, yet at $N = 64$ their HINT-level rankings are nearly unrelated: execution can remain sensitive to a HINT's semantic content even when correcting that HINT in isolation is not beneficial. Native HINT prediction error is insufficient to explain this dissociation.

Nevertheless, the explicit HINT interface remains collectively consequential: restoring all HINT trajectories improves $N = 64$ task accuracy by 15.96 percentage points. Surrounding-context interventions further show that the effect of correcting an individual HINT changes more between the two surrounding HINT states at the larger input size. These results indicate that size extrapolation does not simply eliminate dependence on semantic intermediate states; instead, semantic dependence and isolated repairability remain distinct, while individual correction effects vary more across surrounding HINT states at $N = 64$.

1 INTRODUCTION

Neural Algorithmic Reasoning (NAR) studies whether neural networks can learn to execute classical algorithms and generalize that computation to inputs larger than those observed during training (Veličković & Blundell, 2021). Such larger-size out-of-distribution (OOD) generalization is a central challenge in NAR. The CLRS Algorithmic Reasoning Benchmark provides, for a diverse collection of classical algorithms, inputs, outputs, and trajectories of intermediate variables, referred to as *HINTs*, that evolve throughout algorithm execution (Veličković et al., 2022). Their named semantics and exact ground-truth trajectories make HINTs both explicit supervision targets and a controlled interface for intervention.

However, predicting an intermediate state is not equivalent to establishing that the learned computation causally depends on it (Geiger et al., 2021; Heimersheim & Nanda, 2024). Prior work has shown both that intermediate trajectories can be exploited to improve larger-size OOD generalization and that competitive algorithmic reasoning can be obtained without explicit trajectory supervision (Bevilacqua et al., 2023; Rodionov & Prokhorenkova, 2023). This leaves a basic question unresolved: what causal role do supervised HINTs play during neural execution, and how does that role change under size extrapolation? Crucially, even within a causal intervention framework, asking whether a HINT "matters" is not a single question: correcting it, replacing its input-specific

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.

semantic value, and changing the surrounding state under which it is corrected probe different counterfactuals.

We therefore separate three causal questions about the same semantic HINT variable. For an input size N and HINT h, the *isolated restoration effect* $R_N(h)$ measures the change in task accuracy when only h is replaced by its ground-truth trajectory from the same example. The *semantic-resampling effect* $D_N(h)$ measures the decrease in task accuracy when h is instead replaced by a valid value of the same semantic HINT obtained from another example (Geiger et al., 2021). Finally, we compare the effect of correcting h when the remaining HINTs are model-predicted with its effect when the remaining HINTs are all ground truth. The resulting interaction contrast measures how the effect of the same target-HINT correction changes with the surrounding HINT state. These quantities are not interchangeable measures of HINT importance; they answer distinct counterfactual questions about the same semantic variable.

Across 28 CLRS algorithms with PGN and Triplet-GMPNN processors, both restoration and semantic resampling show positive cross-size rank correspondence. The isolated restoration effect yields $\rho^R_{16,64} = 0.483$ (95% CI $[0.382, 0.588]$), while the semantic-resampling effect, evaluated on HINTs satisfying the source-availability criterion, yields $\rho^D_{16,64} = 0.692$ (95% CI $[0.510, 0.843]$). Thus, the relative ordering of HINT-level intervention effects is partially preserved across input size.

Yet restoration and semantic resampling do not identify the same HINTs at larger size. At $N = 64$, their rank association is $\rho^{R,D}_{64} = -0.035$ (95% CI $[-0.213, 0.149]$). More directly, among HINTs with $R_N(h) < 0$, the fraction with $D_N(h) > 0$ is 97.6% at $N = 16$ and remains 84.3% at $N = 64$. A HINT can therefore remain semantically consequential to the current execution even when making that HINT exactly correct in isolation decreases final accuracy. Native HINT prediction error and correction headroom are insufficient to account for this dissociation.

Nor does this dissociation imply that the HINT interface becomes irrelevant as a whole. Keeping the model weights and inputs fixed while replacing the entire recurrent HINT trajectory with ground truth increases $N = 64$ task accuracy by 15.96 pp (95% CI $[11.14, 21.13]$). The trained network therefore retains substantial capacity to exploit exact intermediate trajectories during larger-size execution.

Finally, the effect of correcting an individual HINT changes with the surrounding HINT state. Let $C_N(h)$ denote the contribution of h when all other HINTs are ground truth, and define the interaction contrast

$$I_N(h) = C_N(h) - R_N(h).$$

The mean magnitude of this contrast increases from $N = 16$ to $N = 64$ by $\Delta_{64-16}|I| = 2.69$ pp (95% CI $[1.45, 4.07]$), showing that the effect of the same correction varies more between the two surrounding HINT states at the larger input size.

Taken together, larger-size OOD does not simply eliminate dependence on semantic intermediate states. The same semantic HINTs retain cross-size structure under both restoration and semantic resampling, yet the two interventions cease to rank HINTs similarly at $N = 64$, and the effect of correcting a HINT varies more across the two surrounding HINT states. Whether a HINT "matters" therefore cannot be reduced to a single context-independent intervention effect: isolated restoration and semantic dependence answer distinct causal questions about learned execution.

2 Related Work

2.1 Intermediate states and OOD generalization in NAR

The CLRS Algorithmic Reasoning Benchmark provides inputs, outputs, and time-varying intermediate HINT trajectories for classical algorithms, and evaluates neural reasoners on OOD generalization to larger input sizes (Veličković et al., 2022). Subsequent work has investigated how such intermediate computation relates to generalization (Ibarz et al., 2022). Hint-ReLIC exploits structure in intermediate transitions to encourage invariant representations and improve larger-size OOD performance (Bevilacqua et al., 2023). In contrast, Rodionov & Prokhorenkova (2023) show that competitive performance on multiple CLRS tasks can be achieved without explicit supervision of algorithmic trajectories. Other analyses of recurrent latent trajectories have identified OOD failure

modes associated with representation resolution and values that move outside the training range (Mirjanic et al., 2024).

These works study the supervision, representation, and generalization of intermediate computation, but do not directly measure how the causal role of a supervised semantic HINT changes with input size. Rather than introducing a new training objective, we intervene on the explicit HINT trajectories of already-trained neural algorithmic reasoners and analyze their causal effects across input sizes.

2.2 Causal interventions for internal computation

Causal tracing restores selected internal states to clean values and measures the resulting recovery in behavior (Meng et al., 2022). Interchange interventions replace an internal representation in a base example with a value obtained from a source example in order to test its counterfactual causal role (Geiger et al., 2021). Work on activation patching further emphasizes that the measured intervention effect can depend on the replacement distribution and evaluation metric, and therefore should not automatically be interpreted as an intrinsic importance score (Heimersheim & Nanda, 2024).

Circuit-level mechanistic interpretability additionally evaluates whether identified components are sufficient or necessary for the target behavior rather than relying on attribution alone (Conmy et al., 2023; Heimersheim & Nanda, 2024). Recent work such as MINAR applies attribution patching to neural algorithmic reasoning and analyzes neuron-level circuit formation and reuse (He et al., 2026). Our analysis operates at a different level of abstraction: the intervention units are not latent neurons or discovered circuits, but the semantic HINT variables explicitly specified by CLRS (Veličković et al., 2022). This allows us to apply multiple causal questions to the same named algorithmic state and examine how its causal effects change from the source size to larger-size OOD execution.

3 Isolated Restoration Reveals Cross-Size Structure and Effect Expansion

3.1 Isolated restoration effect

Let $\mathcal{H}_a$ denote the set of semantic HINT variables for algorithm a, let $h \in \mathcal{H}_a$ denote one such HINT, and let $N \in \{16, 64\}$ denote the input size. We write A_N^{ID} for task accuracy under the unmodified execution and $A_N^{\mathrm{GT}(h)}$ for task accuracy when the recurrent trajectory of h alone is replaced by its ground-truth trajectory from the same example. All other HINTs remain model-predicted.

We define the isolated restoration effect as

$$R_N(h) = 100 \left(A_N^{\mathrm{GT}(h)} - A_N^{\mathrm{ID}} \right).$$

Thus, $R_N(h)$ measures the change in final task accuracy caused by correcting h while leaving all remaining HINTs model-predicted. Because task accuracy is represented as a fraction, $R_N(h)$ is reported in percentage points. A positive $R_N(h)$ indicates that restoring h improves final task accuracy, whereas a negative value indicates that the same intervention decreases it.

We evaluate 28 CLRS algorithms with PGN (Veličković et al., 2020) and Triplet-GMPNN (Ibarz et al., 2022) processors, using seeds 43–45 at $N = 16$ and $N = 64$. Intervention targets are matched by their CLRS semantic variable names (Veličković et al., 2022) rather than by runtime position. For each algorithm–processor pair, the named HINT set used for cross-size comparison must match exactly between $N = 16$ and $N = 64$; all 56 pairs satisfy this requirement.

Algorithms are the primary unit of statistical inference. Seeds 43–45 are first averaged within each algorithm–processor–HINT condition, HINT-level statistics are computed separately for each processor, and the two processor-level values are then averaged within each algorithm. Confidence intervals are obtained by bootstrapping algorithms, so HINTs, seeds, and processors are not treated as independent samples. The resulting inference concerns consistency across the evaluated CLRS algorithms, not population-level generalization to arbitrary NAR architectures.

3.2 RESTORATION ORDERING SHOWS POSITIVE CROSS-SIZE CORRESPONDENCE

For each algorithm–processor pair, we compare the rankings of $R_N(h)$ across HINTs shared between $N = 16$ and $N = 64$. This measures whether HINT-level restoration ordering is preserved across sizes, not whether $R_N(h)$ predicts $N = 64$ task accuracy.

Across the 25 algorithms for which a valid rank correlation can be computed, $\rho^R_{16,64} = 0.483$ (95% CI $[0.382, 0.588]$).

Thus, HINTs with relatively large $R_{16}(h)$ tend to retain relatively large $R_{64}(h)$, indicating non-trivial continuity in restoration structure across size. At the same time, the correspondence is far from complete, indicating substantial reordering of HINT-level restoration effects at the larger size.

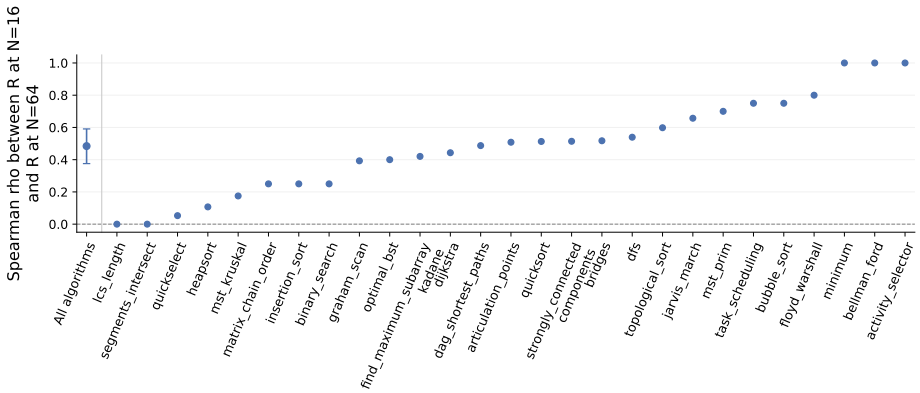

Figure 1: Cross-size restoration correspondence. Algorithm-level Spearman correlation between HINT-level R_{16} and R_{64}; the top row shows the mean with its 95% confidence interval.

3.3 RESTORATION EFFECTS BECOME LARGER AND MORE DISPERSED AT THE LARGER SIZE

Cross-size rank correspondence does not reveal how restoration-effect magnitude changes with size. Without introducing an arbitrary activity threshold, we summarize $R_N(h)$ within each algorithm and processor by its mean absolute effect and standard deviation across HINTs.

The mean absolute restoration effect increases from 2.30 pp at $N = 16$ to 5.67 pp at $N = 64$. The paired algorithm-level increase is 3.37 pp (95% CI $[1.94, 4.82]$), and the increase is observed in 24 of the 28 algorithms.

The spread across HINTs increases as well. The within-algorithm standard deviation of $R_N(h)$ rises from 2.66 pp at $N = 16$ to 6.55 pp at $N = 64$. The corresponding paired algorithm-level increase is 3.89 pp (95% CI $[2.33, 5.54]$), with an increase in 25 of the 28 algorithms.

Hence, larger-size OOD execution is associated with two simultaneous properties. The HINT-level restoration ordering retains non-trivial cross-size structure, yet the effects themselves move farther from zero and become substantially more heterogeneous across HINTs.

A large $R_N(h)$ shows that supplying the correct HINT value can strongly affect final behavior, but does not establish that native execution depends on its input-specific semantic value. We test this distinction next by replacing the same HINT with another valid semantic value rather than its ground truth.

4 SEMANTIC DEPENDENCE PERSISTS ACROSS SIZE BUT DIVERGES FROM ISOLATED RESTORATION

4.1 SEMANTIC-RESAMPLING EFFECT

To distinguish the effect of correcting a HINT from sensitivity to its input-specific semantic content, we replace the target HINT with a different valid value rather than with its ground truth, and denote the resulting task accuracy by $A_N^{\mathrm{SRC}(h)}$.

At each intervened recurrent step, the value of h is replaced by a valid value of the same semantic HINT obtained from another example. All other HINTs remain model-predicted. This replacement follows the general logic of interchange interventions, in which an internal state is replaced by a value obtained from another example (Geiger et al., 2021).

The replacement is neither random noise nor a zero ablation. It remains a valid value of the same named CLRS variable (Veličković et al., 2022), but no longer matches the current input.

We define the semantic-resampling effect as

$$D_N(h) = 100 \left(A_N^{\mathrm{ID}} - A_N^{\mathrm{SRC}(h)} \right).$$

We report $D_N(h)$ in percentage points; a positive value indicates that replacing the input-matched semantic value of h with another valid value decreases final task accuracy.

Restoration and semantic resampling therefore ask different questions. $R_N(h)$ measures the effect of supplying the correct value of h in the current predicted-HINT context, whereas $D_N(h)$ measures sensitivity to replacing the input-specific semantic value carried by h with another valid value of the same HINT. We use semantic dependence operationally to refer to this sensitivity. The two effects should therefore not be interpreted as interchangeable estimates of a single notion of HINT importance (Heimersheim & Nanda, 2024).

We retain only HINTs for which a mismatched valid source value is available for at least 90% of evaluated cases for every seed. The aggregation protocol is otherwise unchanged: seeds 43-45 are averaged for each combination of algorithm, processor, and HINT, HINT-level statistics are computed separately for each processor, and confidence intervals are obtained by bootstrapping algorithms.

4.2 SEMANTIC DEPENDENCE RETAINS CROSS-SIZE STRUCTURE

For each algorithm-processor pair, we compare the rankings of $D_N(h)$ across HINTs valid at both $N = 16$ and $N = 64$. This measures preservation of cross-size semantic-resampling ordering, not the relationship between $D_N(h)$ and $N = 64$ task accuracy.

Across the 20 algorithms for which a valid rank correlation can be computed,

$$\rho^D_{16,64} = 0.692 \qquad (95\%\ \mathrm{CI}\ [0.510, 0.843]).$$

Thus, HINTs with relatively large $D_{16}(h)$ tend to retain relatively large $D_{64}(h)$, indicating substantial continuity in semantic dependence across size.

The correspondence remains positive after excluding HINTs whose names directly match output variables,

$$\rho = 0.625 \qquad (95\%\ \mathrm{CI}\ [0.392, 0.810]),$$

across 15 valid algorithms.

Both isolated restoration and semantic resampling therefore retain non-trivial cross-size structure. The semantic-resampling correspondence is numerically larger, but because the two correlations are computed over different sets of valid algorithms, we do not interpret this difference as evidence of greater cross-size stability.

4.3 Semantic dependence and isolated repairability separate at larger size

Because cross-size structure does not imply that restoration and semantic resampling rank HINTs in the same way, we compare $R_N(h)$ and $D_N(h)$ within each input size.

At $N = 16$, their rank association is weakly positive,

$$\rho_{16}^{R,D} = 0.218 \qquad (95\%\ \text{CI}\ [0.041, 0.393]).$$

At $N = 64$, the association is essentially absent,

$$\rho_{64}^{R,D} = -0.035 \qquad (95\%\ \text{CI}\ [-0.213, 0.149]).$$

A geometry-controlled robustness analysis, detailed in the Appendix, gives a partial correlation of $\rho = -0.218$ (95% CI $[-0.493, 0.043]$, $n = 13$ algorithms) at $N = 64$ and therefore does not remove the observed dissociation.

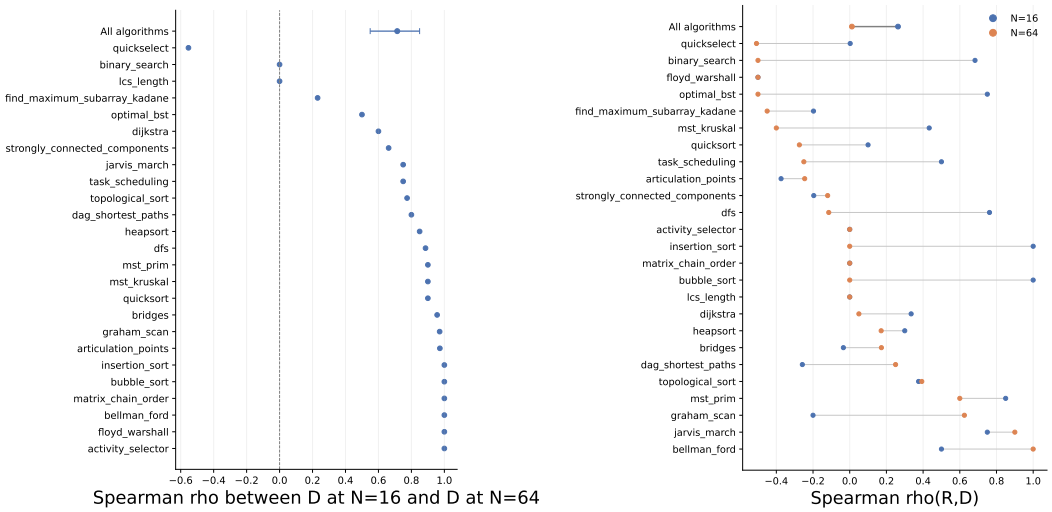

Figure 2: Semantic dependence and its relation to isolated restoration. Left: algorithm-level cross-size Spearman correlation of D. Right: within-size $\rho(R, D)$ at $N = 16$ and $N = 64$.

A HINT may carry semantic information on which the current execution depends while still being ineffective, or even harmful, when corrected in isolation. Conversely, a large restoration effect does not by itself establish that the native execution relies on the input-specific semantic value of that HINT. We therefore treat restoration and semantic resampling as separate causal probes rather than as alternative estimates of a single HINT-importance score.

One possible explanation is correction headroom. If the model already predicts a HINT close to its ground truth, restoration may have little room to improve final accuracy even when execution remains sensitive to its semantic value; we test this explanation next.

5 Prediction Error Is Insufficient to Explain the Restoration–Resampling Dissociation

Restoration and semantic resampling diverge, but the dissociation could arise if some HINTs are already predicted close to ground truth, leaving little room for an exact correction to improve the final output. We test this correction-headroom explanation first, and then examine the more direct case in which ground-truth restoration is harmful despite positive semantic dependence.

5.1 Native HINT Prediction Error Is Insufficient to Explain the Separation

If limited correction headroom were sufficient to explain the dissociation, larger native prediction errors should be associated with larger beneficial restoration effects after differences in semantic

dependence are taken into account. Because this test concerns beneficial restoration and disruptive semantic replacement, we use $R_N^+(h) = \max(R_N(h), 0)$ and $D_N^+(h) = \max(D_N(h), 0)$, and compute the partial Spearman correlation between native HINT prediction error and $R_N^+(h)$ while controlling for $D_N^+(h)$.

The partial association is close to zero at both input sizes: at $N = 16$, $\rho = -0.091$ (95% CI $[-0.372, 0.211]$, $n = 8$ algorithms), and at $N = 64$, $\rho = -0.076$ (95% CI $[-0.276, 0.112]$, $n = 9$ algorithms). A complementary held-out analysis likewise shows no consistent improvement from adding native HINT prediction error: mean absolute error changes from 0.254 to 0.255, while predicted-versus-observed Spearman correlation changes from 0.185 to 0.196.

These results do not imply that native prediction error is irrelevant to individual HINTs. Rather, native prediction error is insufficient to explain the systematic separation between semantic dependence and isolated restoration.

5.2 Negative Restoration Does Not Imply Absence of Semantic Dependence

The distinction is more direct for HINTs with negative restoration effects. Here, $R_N(h) < 0$ means that replacing the predicted trajectory of h with its exact ground-truth trajectory decreases final task accuracy, whereas $D_N(h) > 0$ means that replacing the same HINT with a valid but input-mismatched semantic value also decreases accuracy. Thus, $R_N(h) < 0$ and $D_N(h) > 0$ identify cases in which execution is sensitive to preserving the input-specific semantic content of h, even though correcting that HINT in isolation is harmful.

Such cases are not rare. Among HINTs with negative restoration effects, the fraction that nevertheless satisfy $D_N(h) > 0$ is 97.6% at $N = 16$ (95% CI $[94.4\%, 99.8\%]$, $n = 23$ algorithms) and 84.3% at $N = 64$ (95% CI $[70.0\%, 96.2\%]$, $n = 21$ algorithms).

The corresponding semantic-dependence effects are substantial. Averaging the within-algorithm median $D_N(h)$ among HINTs with negative restoration effects gives 11.96 pp at $N = 16$ (95% CI $[7.88, 16.43]$) and 8.08 pp at $N = 64$ (95% CI $[4.10, 12.13]$).

Harmful ground-truth restoration therefore does not imply that the corresponding semantic information is unused by native execution, and semantic dependence does not imply that making one HINT exactly correct in isolation will improve final behavior. This result does not identify why ground-truth restoration can be harmful, but it directly establishes that sensitivity to a HINT's semantic content and the benefit of correcting that HINT in isolation are distinct properties of neural algorithmic execution.

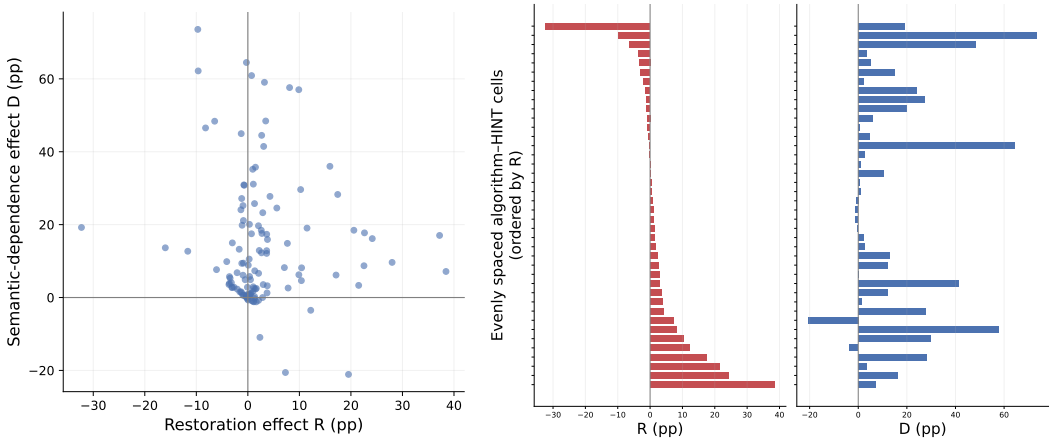

Figure 3: Restoration–semantic-dependence dissociation at $N = 64$. Left: HINT-level R versus D. Right: algorithm–HINT cells sampled at evenly spaced ranks after ordering by R, with the corresponding D values shown in the same order.

6 SEMANTIC HINTS REMAIN CONSEQUENTIAL BUT BECOME MORE CONTEXT-DEPENDENT

A negative restoration effect does not imply that the corresponding HINT is behaviorally irrelevant. We therefore ask whether the HINT interface remains collectively consequential at larger sizes and whether the effect of an individual HINT becomes more dependent on the state of the remaining HINTs.

6.1 CORRECT HINT TRAJECTORIES REMAIN COLLECTIVELY CONSEQUENTIAL AT LARGER SIZE

If the semantic HINT interface remains consequential for execution, replacing the full recurrent HINT trajectory with the ground-truth trajectory should substantially change the final behavior of the frozen model.

We define an oracle condition, A_N^{ALL}, in which every explicit recurrent HINT trajectory is replaced by the ground-truth trajectory for the current input while model parameters and inputs remain unchanged. This intervention does not represent a deployable inference procedure; it measures how much the existing computation can change when exact intermediate semantic states are supplied.

At $N = 16$, accuracy increases from 92.48% to 98.50%, a gain of $+6.02$ percentage points (pp; 95% CI $[2.73, 10.14]$). At $N = 64$, accuracy increases from 67.91% to 83.87%, a substantially larger gain of $+15.96$ pp (95% CI $[11.14, 21.13]$). Thus, the collective influence of the supervised HINT trajectories does not disappear at the larger size. However, accuracy remains 83.87% under the all-ground-truth condition at $N = 64$, leaving 16.13 pp of error. Exact HINT trajectories therefore provide substantial collective leverage but are not sufficient for correct larger-size execution. A small or negative isolated restoration effect therefore does not imply that the semantic state interface as a whole is unimportant.

6.2 INDIVIDUAL HINT EFFECTS BECOME MORE CONTEXT-DEPENDENT AT LARGER SIZE

To test this possibility, we keep the target intervention fixed and vary only the state of the remaining HINTs. If correcting the same target HINT h produces different effects when the other HINTs are model predictions versus ground truth, then the effect of h depends on its surrounding HINT context.

Let $A_N^{\mathrm{ALL}\setminus\{h\}}$ denote the condition in which every HINT except h is replaced by its ground-truth trajectory, while h remains the model prediction. The contribution of h within this all-ground-truth surrounding context is

$$C_N(h) = 100\left(A_N^{\mathrm{ALL}} - A_N^{\mathrm{ALL}\setminus\{h\}}\right). \tag{1}$$

The isolated restoration effect $R_N(h)$ and the conditional effect $C_N(h)$ apply the same correction to h under different surrounding HINT states. Their difference,

$$I_N(h) = C_N(h) - R_N(h), \tag{2}$$

therefore measures how much the target-HINT effect changes between the two surrounding states (VanderWeele, 2009). We use $|I_N(h)|$ to measure the magnitude of this change irrespective of direction. This quantity does not identify pairwise interactions between particular HINTs.

To test whether individual HINT effects change more with the surrounding state at larger size, we compare the average magnitude of this contrast between $N = 16$ and $N = 64$. Across 28 algorithms, we observe

$$\mathrm{mean}\,|I_{64}| - \mathrm{mean}\,|I_{16}| = +2.69\ \mathrm{pp}, \tag{3}$$

with an algorithm-level bootstrap 95% CI of $[1.45, 4.07]$ pp. Individual HINT effects therefore change more strongly between the two surrounding HINT states at $N = 64$ than at $N = 16$.

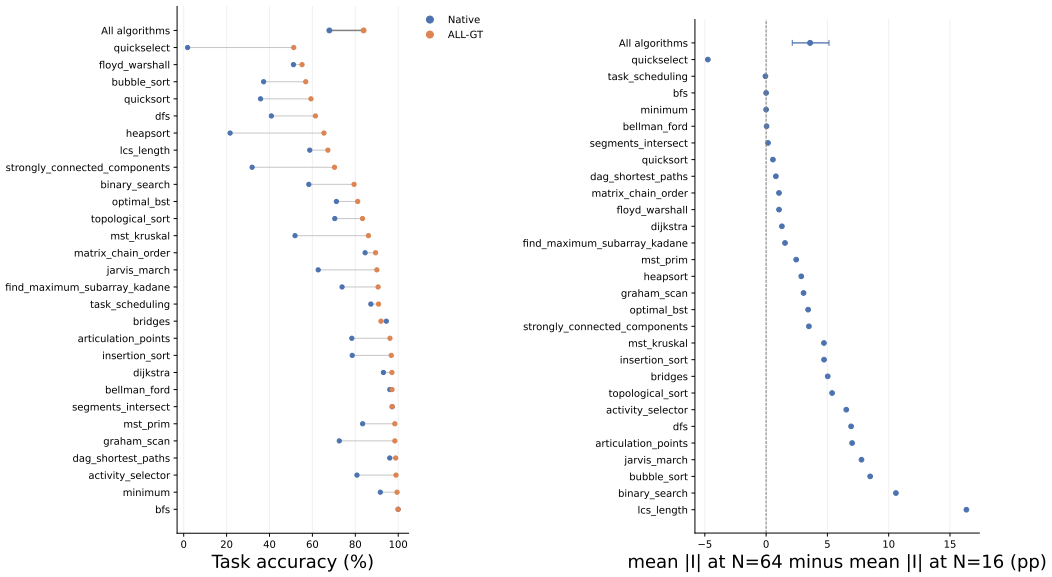

Figure 4: Collective HINT influence and context sensitivity. Left: native versus ALL-GT task accuracy at $N = 64$. Right: algorithm-level change in mean $|I|$ from $N = 16$ to $N = 64$.

6.3 Correcting the surrounding HINTs does not reconcile restoration with semantic dependence

Context sensitivity alone does not explain why restoration and semantic dependence can disagree. A particularly simple explanation is that isolated restoration creates an inconsistent state: the target HINT is replaced by its ground-truth value while the remaining HINTs retain the model's predictions. Under this explanation, a correct value could be harmful not because the model fails to use its semantic content, but because that value is introduced into an incompatible surrounding state.

This explanation makes two direct predictions. First, once the surrounding HINTs are also corrected, the conditional effect $C_N(h)$ should align more closely with semantic dependence $D_N(h)$ than the isolated restoration effect $R_N(h)$ does. Second, HINTs for which $R_N(h) < 0$ but $D_N(h) > 0$ should frequently become beneficial, $C_N(h) > 0$, when evaluated within the all-ground-truth surrounding context.

The first prediction receives little support at $N = 64$: $\rho(C, D) - \rho(R, D) = +0.070$ (95% CI $[-0.222, 0.360]$, $n = 16$ algorithms), so replacing the surrounding HINTs with ground truth does not produce a clear realignment between restoration-based and semantic-resampling-based rankings.

The second prediction is also only partially supported. Among algorithms containing HINTs with $R_N(h) < 0$ and $D_N(h) > 0$, the fraction for which the effect becomes positive under the all-ground-truth surrounding context is 39.3% (95% CI $[23.9, 55.6]$, $n = 19$ algorithms), so correcting the surrounding HINTs rescues some negative-restoration cases, but not most of them.

These results do not support mixed ground-truth/predicted-state inconsistency as a sufficient explanation for the restoration–semantic-dependence dissociation. The surrounding state modifies individual intervention effects, yet placing the target HINT in an all-ground-truth context does not collapse restoration and semantic dependence into the same ordering.

Taken together, the same semantic HINTs retain substantial cross-size structure, but whether a HINT "matters" cannot be reduced to a single context-independent intervention effect. The HINT interface remains collectively consequential at larger size, while individual correction effects vary more between the two surrounding HINT states at $N = 64$. These results do not establish context dependence as the cause of larger-size OOD failure. Semantic dependence and isolated repairability are nevertheless distinct properties: a model may depend strongly on the information carried by a HINT without making that HINT an independently repairable component of its learned computation.

AI USE STATEMENT

Generative AI tools were used to assist with hypothesis refinement, experimental design, implementation and debugging of experimental and analysis code, result interpretation, literature search, visualization, and drafting and editing of the manuscript.

All training, interventions, evaluations, and quantitative analyses reported in this work were executed in the author's research environment. The reported numerical results were obtained from the resulting experimental outputs rather than generated by the language model.

The author reviewed the experimental and analysis code underlying the reported results, including the intervention logic and statistical aggregation, checked reported quantities against the underlying outputs, and verified that the figures reflect the corresponding result data. AI-assisted interpretations and manuscript text were also reviewed against the experimental evidence and cited literature.

The author made the final decisions regarding experimental validity, interpretation, and presentation, and takes responsibility for the full content of the submission.

REPRODUCIBILITY STATEMENT

The main text specifies the intervention definitions, evaluation settings, statistical aggregation, and primary estimands used in Sections 3–6. The Appendix provides the full experimental configuration, semantic-resampling donor construction and coverage criteria, intervention implementation details, aggregation and bootstrap procedures, integrity checks, and additional robustness analyses. We additionally provide anonymized supplementary code for reproducing the reported interventions, analyses, and figures, together with the corresponding experiment configurations. All experiments use the public CLRS benchmark and the model, seed, and input-size settings documented in the paper and Appendix.

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 *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2272–2288. PMLR, 2023.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. In *Advances in Neural Information Processing Systems*, volume 36, pp. 16318–16352, 2023. doi: 10.52202/075280-0719.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 9574–9586, 2021.

Jesse He, Helen Jenne, Max Vargas, Davis Brown, Gal Mishne, Yusu Wang, and Henry Kvinge. MINAR: Mechanistic interpretability for neural algorithmic reasoning. arXiv preprint arXiv:2602.21442, 2026. URL https://arxiv.org/abs/2602.21442.

Stefan Heimersheim and Neel Nanda. How to use and interpret activation patching. arXiv preprint arXiv:2404.15255, 2024. URL https://arxiv.org/abs/2404.15255.

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 *Proceedings of the First Learning on Graphs Conference*, volume 198 of *Proceedings of Machine Learning Research*, pp. 2:1–2:23. PMLR, 2022.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-1262.

Vladimir V. Mirjanic, Razvan Pascanu, and Petar Veličković. Latent space representations of neural algorithmic reasoners. In *Proceedings of the Second Learning on Graphs Conference*, volume 231 of *Proceedings of Machine Learning Research*, pp. 10:1–10:24. PMLR, 2024.

Gleb Rodionov and Liudmila Prokhorenkova. Neural algorithmic reasoning without intermediate supervision. In *Advances in Neural Information Processing Systems*, volume 36, pp. 51663–51674, 2023. doi: 10.52202/075280-2249.

Tyler J. VanderWeele. On the distinction between interaction and effect modification. *Epidemiology*, 20(6):863–871, 2009. doi: 10.1097/EDE.0b013e3181ba333c.

Petar Veličković and Charles Blundell. Neural algorithmic reasoning. *Patterns*, 2(7):100273, 2021. doi: 10.1016/j.patter.2021.100273.

Petar Veličković, Lars Buesing, Matthew Overlan, Razvan Pascanu, Oriol Vinyals, and Charles Blundell. Pointer graph networks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 2232–2244, 2020.

Petar Veličković, Adrià Puigdomènech Badia, David Budden, Razvan Pascanu, Andrea Banino, Misha Dashevskiy, Raia Hadsell, and Charles Blundell. The CLRS algorithmic reasoning benchmark. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22084–22102. PMLR, 2022.

A Experimental Details

A.1 Benchmark and Experimental Cohort

We use the CLRS Algorithmic Reasoning Benchmark (Veličković et al., 2022) through the official `dm-clrs` implementation. The upstream training backbone covers all 30 CLRS algorithms using two neural processors, PGN (Veličković et al., 2020) and Triplet-GMPNN (Ibarz et al., 2022), with training seeds 42–45.

For non-string algorithms, procedural training cycles over the nominal input sizes

$$N \in \{4, 7, 11, 13, 16\}.$$

The two CLRS string-matching tasks, `kmp_matcher` and `naive_string_matcher`, use the benchmark-specific string length convention and do not share this common five-size training schedule. They were therefore not included in the pre-specified 28-algorithm cohort used for the subsequent intervention study. This exclusion was fixed before the intervention outcomes were examined.

The intervention experiments therefore use the following 28 algorithms: `articulation_points`, `activity_selector`, `bellman_ford`, `bfs`, `binary_search`, `bridges`, `bubble_sort`, `dag_shortest_paths`, `dfs`, `dijkstra`, `find_maximum_subarray_kadane`, `floyd_warshall`, `graham_scan`, `heapsort`, `insertion_sort`, `jarvis_march`, `lcs_length`, `matrix_chain_order`, `minimum`, `mst_kruskal`, `mst_prim`, `optimal_bst`, `quickselect`, `quicksort`, `segments_intersect`, `strongly_connected_components`, `task_scheduling`, and `topological_sort`.

All primary intervention analyses use the step-10,000 checkpoints from seeds 43–45 and evaluate the same two input sizes, $N = 16$ and $N = 64$. Model parameters remain frozen throughout all intervention experiments.

A.2 Models and Training

Both processors are instantiated using the CLRS `BaselineModel`. HINTs are explicitly encoded and decoded, and predicted HINTs are recurrently fed back using the `soft` reprediction mode. The common hidden dimension is 128, LayerNorm is enabled, dropout is zero, LSTM recurrence is disabled, and one message-passing step is applied per recurrent execution step.

The common and processor-specific settings are summarized in Table 1. The processor-specific HINT teacher-forcing rates were inherited from the frozen upstream training recipe and were not tuned for the intervention study: PGN uses a probability of 0.5, whereas Triplet-GMPNN uses no HINT teacher forcing (0.0). Triplet-GMPNN uses eight triplet features. Models are trained for 10,000 optimizer updates using Adam with learning rate 10^{-3}, batch size 32, and gradient clipping at maximum norm 1.0.

Table 1: Training and model configuration of the frozen training backbone used to obtain the intervention checkpoints.

Setting	Value
Processors	PGN, Triplet-GMPNN
Training seeds	42, 43, 44, 45
Training updates	10,000
Batch size	32
Hidden dimension	128
Optimizer	Adam
Learning rate	10^{-3}
Gradient clipping	1.0
LayerNorm	enabled
Dropout	0.0
LSTM	disabled
Message-passing steps	1
HINT reprediction	soft
PGN HINT teacher forcing	0.5
Triplet-GMPNN HINT teacher forcing	0.0
Triplet features	8

Random positional features and the CLRS preprocessing procedures for predicted-as-input variables and permutations are applied according to the corresponding task specification. Fixed larger-size evaluations recorded during training are observational only: their scores are not used for optimization, early stopping, checkpoint selection, probe selection, batch-size adaptation, or algorithm-specific tuning. The paper analyses use the checkpoint fixed by the 10,000-update training schedule rather than selecting a checkpoint using larger-size performance.

A.3 Evaluation Protocol

For each algorithm and input size, evaluation uses a fixed deterministic dataset generated independently of the training sampler and training random-number stream. The same evaluation instances are shared across processors and seeds for a given algorithm and input size.

Each evaluation contains 64 examples. At $N = 16$, these are processed as two batches of 32 examples, and at $N = 64$ as eight batches of eight examples. Dataset-stream hashes are stored with the frozen evaluation cache. Before an intervention is run, the corresponding evaluation stream is reconstructed and its hash is checked against the frozen reference; a mismatch causes the run to terminate.

Native evaluation uses free-running recurrent inference, with predicted HINTs fed back at each execution step. Ground-truth HINT trajectories are not injected during native evaluation. For task evaluation, predictions and targets from all 64 examples are first concatenated along the batch dimension, after which the official `clrs.evaluate` function is called once. This avoids averaging nonlinear evaluation metrics independently across mini-batches.

All intervention conditions use the same frozen checkpoint and evaluation examples as their corresponding native condition. No optimizer or parameter update is performed during intervention evaluation. The exact construction of the restoration, semantic-resampling, and surrounding-context interventions is described in Appendix B.

A.4 Analysis Cohorts and Statistical Aggregation

We treat each algorithm as the unit of analysis for statistical inference, rather than treating individual HINTs, seeds, or processors as independent samples. For the primary causal analyses, seeds 43–45

are first averaged within each algorithm–processor–HINT condition. HINT-level statistics are then computed separately for PGN and Triplet-GMPNN, and the two processor-level values are averaged within each algorithm before results are summarized across algorithms.

For semantic-resampling analyses, we retain only HINTs for which a valid mismatched source value is available for at least 90% of evaluated cases for every seed. Cross-size rank comparisons use only HINTs that are valid at both $N = 16$ and $N = 64$. For the R and D rank analyses, a Spearman correlation is computed for an algorithm–processor pair only when at least three finite paired HINTs are available and neither quantity is constant. Any additional filtering used for a particular robustness analysis is stated together with that analysis.

Unless stated otherwise, we compute 95% bootstrap confidence intervals by resampling algorithms with replacement. HINTs, seeds, and processors are not resampled as independent observations. The resulting confidence intervals therefore measure how consistently the reported effects hold across algorithms, rather than across individual HINTs or random seeds.

A.5 Implementation and Compute Environment

Core experiment metadata record Python 3.11.15, `dm-clrs` 2.0.3, JAX 0.4.31, and JAXLIB 0.4.31 with the GPU backend. The reproduction environment used for the final audit additionally uses Haiku 0.0.12 and Optax 0.2.3.

Training and intervention experiments are parallelized across two GPUs using a shared job queue. Each job corresponds to one algorithm–processor–seed configuration and runs entirely on a single GPU. Jobs follow a predetermined queue order; whenever a GPU finishes its current job, it takes the next pending job. Jobs are therefore not statically assigned to a particular GPU, and a single model run is never split across multiple GPUs.

The current reproduction workstation contains an NVIDIA GeForce RTX 4070 Ti SUPER and an NVIDIA GeForce RTX 4080, each with 16 GB of memory. The complete dependency and environment record is retained with the supplementary material.

B Intervention Protocol and Validity Checks

B.1 Intervention Design and Recurrent Execution

All interventions are performed on the explicit recurrent HINT interface of a frozen model. For a target HINT h, the five evaluation conditions used throughout the paper are summarized in Table 2.

Table 2: Intervention conditions used to define the causal effects in the main text. Predicted values are the model's own recurrent HINT predictions, own GT denotes the ground-truth HINT trajectory of the current example, and valid source denotes a valid value of the same HINT obtained from another example.

Condition	Target HINT h	Remaining HINTs
A_N^{ID}	predicted	predicted
$A_N^{\mathrm{GT}(h)}$	own GT	predicted
$A_N^{\mathrm{SRC}(h)}$	valid source	predicted
$A_N^{\mathrm{ALL}\setminus\{h\}}$	predicted	own GT
A_N^{ALL}	own GT	own GT

The identity condition leaves all recurrent HINT values unchanged. For isolated restoration, only the target HINT is replaced by its ground-truth trajectory from the current example. For semantic resampling, only the target HINT is replaced by a valid source value from another example. The all-ground-truth condition replaces every recurrent HINT with its ground-truth trajectory, while the leave-one-out condition leaves the target HINT model-predicted and replaces all remaining HINTs with ground truth. These conditions define $R_N(h)$, $D_N(h)$, $C_N(h)$, and $I_N(h)$ as specified in the main text.

The replacements are applied during recurrent execution rather than to the final output after inference. The initialization step is left unchanged. At each subsequent valid feedback step, the relevant HINT value is replaced according to the intervention condition, and the model then continues its recurrent computation with the modified HINT fed back as input. Model parameters remain unchanged throughout.

Interventions are applied only while an example remains active in the recurrent execution. Replacement values are represented in the same form as the HINT values normally fed back to the model. For pointer-valued HINTs, the CLRS ground truth stores a discrete node index, whereas the recurrent model input represents the pointer as a vector over nodes. The ground-truth index is therefore converted to the corresponding one-hot vector before it is fed back. Before evaluation, each replacement is also verified to have the same shape as its corresponding recurrent HINT input.

No intervention directly changes the original problem input, the model parameters, or the final output. Changes in later hidden states and outputs arise through the model's normal recurrent computation after the modified HINT value has been fed back.

B.2 Semantic-Resampling Donor Selection

Semantic resampling follows the source-example replacement principle used in interchange interventions and activation patching (Geiger et al., 2021; Heimersheim & Nanda, 2024). Instead of using random noise or a zero value, the target HINT is replaced by a valid value of the same semantic variable taken from another example.

For each target example and recurrent step, source candidates are taken from other valid examples in the same evaluation batch at the same step. A candidate must correspond to the same named HINT, must still be valid at that step, and must have a ground-truth value different from that of the target example.

Source selection is deterministic. Starting from the next example in the batch and wrapping around when necessary, we select the first example satisfying these conditions. The selection rule depends only on the ground-truth HINT values and example validity at that step; it does not use the model's native prediction, task accuracy, or intervention outcome. Distances between the native prediction, the target ground truth, and the selected source value are recorded separately and examined as a robustness check.

The complete source schedule is fixed before the corresponding intervention outcomes are evaluated. If no valid source example is available for a particular example and recurrent step, no semantic replacement is applied at that step.

For the primary semantic-resampling analyses, a HINT is retained only when a valid source value is available for at least 90% of valid example–steps for every seed. The same requirement is applied before $D_N(h)$ is included in the reported cross-size and restoration–resampling analyses.

B.3 Target Identification and Integrity Checks

The same semantic HINT variables can appear in a different list order during model execution than in the CLRS task specification. We therefore identify each intervention target by its CLRS variable name rather than by its position in the runtime list. Before evaluation, the runtime HINT-name set is checked against the expected set for the corresponding task.

This distinction matters in practice. Across the 336 algorithm–processor–seed–size evaluations in the final targeting audit, all target-name checks passed. The runtime and specification orders were identical in only 12 evaluations and differed in the remaining 324.

We also evaluate an identity condition through the same intervention code path while leaving all HINT values unchanged. Its predictions matched the corresponding native execution in all 1,680 evaluated mini-batches. All selected semantic-resampling donors satisfied the eligibility rules described above.

Each reported causal effect is computed from conditions evaluated within the same intervention implementation. In particular, $R_N(h)$ and $D_N(h)$ use the identity condition evaluated alongside their restoration and resampling conditions, while $C_N(h)$ compares the all-ground-truth and leave-

one-out conditions evaluated together. Absolute native accuracies produced by separate intervention programs are used for consistency checks but are not combined to form these effect estimates.

B.4 Interpretation of the Intervention Measures

The intervention measures answer different counterfactual questions about the same semantic HINT and are not interchangeable measures of HINT importance. $R_N(h)$ measures the effect of correcting the target HINT while the remaining HINTs stay model-predicted. $D_N(h)$ measures sensitivity to replacing the target's input-specific semantic value with another valid value of the same HINT. $C_N(h)$ applies the same target correction while the remaining HINTs are ground truth, and $I_N(h)$ measures how the target-HINT effect changes between these two surrounding HINT conditions.

These interventions characterize causal effects within the explicit HINT interface. They do not show that the HINT variables fully determine the model's computation. The all-ground-truth condition also uses information unavailable during ordinary test-time inference and is therefore an analysis condition rather than an inference method. Likewise, $I_N(h)$ measures how the effect of correcting one HINT changes between two joint states of the remaining HINTs; it does not identify an interaction between a particular pair of HINTs.

C Analysis Details and Robustness Checks

C.1 Native HINT Prediction Error

Because CLRS HINTs have different data types, prediction error is measured using a metric appropriate to each type.

For pointer, categorical, and `MASK_ONE` HINTs, predictions are converted to discrete classes by taking the argmax, and prediction error is defined as one minus classification accuracy. For binary `MASK` HINTs, values are thresholded at 0.5. We use balanced accuracy when both classes are present, because ordinary accuracy can be dominated by the majority class; when balanced accuracy is undefined, we use binary accuracy instead. Prediction error is again one minus the resulting accuracy.

For scalar HINTs, we use root-mean-square error normalized by the standard deviation of the ground-truth values,

$$E_{\text{scalar}}(h) = \frac{\text{RMSE}(h)}{\text{Std}(h_{\text{GT}}) + \epsilon}.$$

Because this quantity is not on the same measurement scale as the classification errors above, scalar HINTs are analyzed separately and are not included in the main prediction-error analysis.

Table 3: Prediction-error measures used for the different CLRS HINT types.

HINT type	Prediction metric	Error used in analysis
Pointer / categorical / `MASK_ONE`	Classification accuracy	$1 - \text{accuracy}$
`MASK`	Balanced accuracy	$1 - \text{balanced accuracy}$
Scalar	Normalized RMSE	Analyzed separately

Seeds 43–45 are first averaged within each algorithm–processor–size–HINT combination. The analyses below use non-scalar HINTs for which a valid semantic-resampling source is available for at least 90% of valid example–steps for every seed.

C.2 Native Prediction Error and Restoration

One possible explanation for a large $D_N(h)$ but a small $R_N(h)$ is that the model already predicts h accurately, leaving little room for a ground-truth correction to improve the final output. If this explanation is sufficient, HINTs with larger native prediction errors should show larger beneficial restoration effects after differences in semantic dependence are accounted for.

Because this question concerns beneficial restoration and disruptive semantic replacement, we use

$$R_N^+(h) = \max(R_N(h), 0), \qquad D_N^+(h) = \max(D_N(h), 0).$$

Within each algorithm–processor–size group, we rank R_N^+, D_N^+, and native prediction error across HINTs. We then compute a partial Spearman correlation between R_N^+ and native prediction error while controlling for D_N^+.

Operationally, the ranks of R_N^+ and native prediction error are each regressed on the rank of D_N^+, and the correlation between the two residuals is measured. This asks whether HINTs with larger native errors also tend to have larger beneficial restoration effects after the association of both quantities with semantic dependence has been removed. The calculation requires at least five eligible HINTs for an algorithm–processor pair.

The processor-level correlations are averaged within each algorithm, and confidence intervals are obtained by bootstrapping algorithms. The resulting partial association is small at both input sizes:

$$\rho = -0.091 \quad (95\%\ \text{CI}\ [-0.372,\ 0.211],\ n = 8)$$

at $N = 16$, and

$$\rho = -0.076 \quad (95\%\ \text{CI}\ [-0.276,\ 0.112],\ n = 9)$$

at $N = 64$.

As a complementary out-of-sample check, we test whether native prediction error improves prediction of restoration ranks for an unseen algorithm. We use leave-one-group-out cross-validation, treating each algorithm as one group. For each split, all rows from one algorithm are held out, linear regression models are fitted by least squares using the remaining algorithms, and prediction is evaluated only on the held-out algorithm.

Before fitting, R_N^+, D_N^+, and native prediction error are converted to ranks and rescaled to $[0, 1]$ within each algorithm–processor–size group. We denote these values by r_R, r_D, and r_E. We compare

$$\widehat{r}_R = \beta_0 + \beta_1 r_D,$$

$$\widehat{r}_R = \beta_0 + \beta_1 r_D + \beta_2 r_E,$$

and

$$\widehat{r}_R = \beta_0 + \beta_1 r_D + \beta_2 r_E + \beta_3 r_D r_E.$$

Here, β_0 is the intercept, β_1 is the coefficient for semantic-resampling rank, β_2 is the additional coefficient for prediction-error rank, and β_3 allows the relationship with prediction error to vary with semantic-resampling rank. The coefficients are re-estimated for every held-out algorithm, so the comparison is based on held-out prediction rather than on any single set of coefficient values.

Table 4: Prediction of restoration rank under leave-one-group-out cross-validation, with algorithms used as groups. Lower mean absolute error and higher Spearman correlation indicate better prediction on held-out algorithms.

Predictors	Mean absolute error	Spearman ρ
r_D	0.254	0.185
r_D, r_E	0.255	0.196
$r_D, r_E, r_D r_E$	0.255	0.188

Adding native prediction error does not produce a consistent improvement in held-out prediction. Mean absolute error slightly increases, the increase in Spearman correlation is small, and adding the interaction term returns the correlation close to the semantic-resampling-only result. Together with the partial-correlation analysis, these results indicate that native prediction error can matter for individual HINTs but is insufficient to explain the systematic separation between semantic dependence and isolated restoration.

C.3 Controlling for Replacement Distance

Another possibility is that a source value has a larger effect simply because it is farther from the model's native prediction than the corresponding ground-truth value. If this difference in distance explains the weak association between $R_N(h)$ and $D_N(h)$, then a positive association between the two effects should reappear after the distance measures are controlled.

The source-selection rule described above does not use the model's native prediction or any distance measure. We therefore measure these distances after the source values have been selected. At each valid example–step, we compute the RMS distance from the native recurrent prediction to the target example's ground truth, d_{GT}, and the RMS distance from the native prediction to the selected source value, d_{SRC}. We also record

$$\frac{d_{\mathrm{GT}}}{d_{\mathrm{GT}} + d_{\mathrm{SRC}} + \epsilon},$$

which represents the relative size of the two distances. These quantities are averaged over the valid example–steps for each HINT.

We then compute the partial rank correlation between $R_N(h)$ and $D_N(h)$ while controlling for the ranks of the two RMS distances and their relative-distance ratio. The same 90% source-availability requirement used for the primary semantic-resampling analysis is applied.

At $N = 64$, the primary restoration–resampling association is

$$\rho_{64}^{R,D} = -0.035 \quad (95\%\ \mathrm{CI}\ [-0.213,\ 0.149],\ n = 20).$$

After controlling for the distance measures, the association is

$$\rho = -0.218 \quad (95\%\ \mathrm{CI}\ [-0.493,\ 0.043],\ n = 13).$$

Thus, accounting for these measured distance differences does not restore a positive association between restoration and semantic resampling at the larger input size.

At $N = 16$, the corresponding partial correlation is

$$\rho = 0.484 \quad (95\%\ \mathrm{CI}\ [0.215,\ 0.722],\ n = 12).$$

This analysis addresses differences in the distance between the native prediction and the replacement values. It does not establish that a source HINT forms a natural combination with all other HINTs of the target example.

C.4 Excluding HINTs that Directly Match Output Variables

A further possibility is that the observed effects are driven mainly by HINTs that correspond directly to task outputs. We identify such HINTs by exact name matching. If a HINT name ends in `_h`, the suffix is removed and the remaining name is compared with the output-variable names in the CLRS task specification.

After excluding HINTs identified in this way, the cross-size rank correspondence of semantic-resampling effects remains positive:

$$\rho_{16,64}^{D} = 0.625 \quad (95\%\ \mathrm{CI}\ [0.392,\ 0.810],\ n = 15).$$

Thus, the cross-size structure of semantic resampling is not limited to HINTs whose names directly match task outputs.

We also repeat the all-ground-truth intervention while leaving these HINTs model-predicted. All other recurrent HINTs are replaced by their ground-truth trajectories. The mean accuracy gain is

$$+4.66\ \mathrm{pp} \quad (95\%\ \mathrm{CI}\ [1.82,\ 8.56])$$

at $N = 16$, and

$$+8.12\ \mathrm{pp} \quad (95\%\ \mathrm{CI}\ [3.03,\ 13.42])$$

at $N = 64$.

For comparison, replacing all recurrent HINTs with ground truth yields an $N = 64$ gain of $+15.96$ pp. HINTs whose names directly match output variables therefore contribute to the full effect, but they do not account for it entirely: a substantial gain remains when those HINTs are left model-predicted.

D Processor- and Algorithm-Level Results

The primary analyses treat the algorithm as the unit of statistical inference and average processor-level statistics within each algorithm. We report the corresponding processor-stratified and algorithm-level results here to make the heterogeneity underlying the aggregate results explicit. These results provide a disaggregated view of the primary analyses and do not define additional inferential claims.

D.1 Processor-Stratified Results

Table 5 reports the principal quantities separately for PGN and Triplet-GMPNN. Confidence intervals are obtained by bootstrapping algorithms within each processor using the same analysis cohorts as the corresponding primary results.

Table 5: Primary results pooled across processors and reported separately for PGN and Triplet-GMPNN. Entries show algorithm-level means with 95% bootstrap confidence intervals. Accuracy differences are in percentage points.

Metric	Pooled	PGN	Triplet-GMPNN
$\rho^{R}_{16,64}$	0.483 [0.382, 0.588]	0.588 [0.444, 0.720]	0.368 [0.199, 0.530]
$\rho^{D}_{16,64}$	0.692 [0.510, 0.843]	0.697 [0.471, 0.888]	0.688 [0.516, 0.827]
$\rho^{R,D}_{64}$	−0.035 [−0.213, 0.149]	0.087 [−0.115, 0.286]	−0.158 [−0.419, 0.107]
$A^{\mathrm{ALL}}_{16} - A^{\mathrm{ID}}_{16}$ (pp)	+6.02 [2.73, 10.14]	+11.88 [5.59, 19.82]	+0.15 [−1.12, 1.06]
$A^{\mathrm{ALL}}_{64} - A^{\mathrm{ID}}_{64}$ (pp)	+15.96 [11.14, 21.13]	+24.36 [16.19, 33.31]	+7.56 [2.60, 13.28]
mean $\lvert I_{64}\rvert$ − mean $\lvert I_{16}\rvert$ (pp)	+2.69 [1.45, 4.07]	+3.89 [1.84, 6.34]	+1.49 [0.54, 2.49]

The main cross-size patterns are visible in both processor strata. Both processors show positive cross-size correspondence for isolated restoration and semantic resampling. At $N = 64$, the association between restoration and semantic resampling is weak in both strata. The all-ground-truth gain is positive for both processors at $N = 64$, and the mean magnitude of the interaction contrast increases from $N = 16$ to $N = 64$ for both processors.

The effect magnitudes are nevertheless processor-dependent. In particular, the all-ground-truth gain at $N = 16$ is substantial for PGN but close to zero for Triplet-GMPNN. The pooled results should therefore be interpreted as structure shared across the two evaluated processors while allowing substantial differences in effect magnitude.

D.2 Algorithm-Level Results and HINT Coverage

Because the algorithm is the primary unit of inference, Table 6 reports the corresponding point estimates for each of the 28 algorithms. The # HINTs column gives the total number of named recurrent HINT variables for each algorithm. It does not give the number of HINTs retained for semantic resampling.

For $\rho^{R}_{16,64}$ and $\rho^{D}_{16,64}$, correlations are first computed separately for PGN and Triplet-GMPNN and the finite processor-level values are then averaged within each algorithm, following the primary aggregation procedure. The all-ground-truth and interaction quantities are likewise aggregated within algorithm.

Across the 28 algorithms, the evaluation cohort contains 164 named recurrent HINT variables, corresponding to 328 processor–HINT pairs at each input size. The semantic-resampling source-availability criterion is satisfied by 262 of 328 processor–HINT pairs at $N = 16$ and 244 of 328 at $N = 64$. A total of 236 of 328 pairs satisfy the criterion at both sizes. No source-eligibility violations are observed.

The difference between the total HINT count and the number of HINTs available for $\rho^{D}_{16,64}$ is important. Semantic resampling requires a valid source value for at least 90% of valid example–steps for every seed, and a HINT must satisfy this requirement at both $N = 16$ and $N = 64$ to enter the cross-size correlation. The correlation additionally requires at least three finite paired HINTs and non-constant ranks. Isolated restoration does not require a source example and therefore does not incur the semantic-resampling coverage filter.

After these validity requirements are applied, $\rho^{D}_{16,64}$ is defined for 20 algorithms and $\rho^{R}_{16,64}$ for 25 algorithms. The dashes in Table 6 therefore represent undefined correlations under the stated analysis criteria rather than missing algorithm evaluations.

The disaggregated results also make clear that the cohort-level effects are not uniform across algorithms. Some algorithms show small or negative all-ground-truth gains, and a small number show decreases rather than increases in mean $|I|$ from $N = 16$ to $N = 64$. The primary conclusions

Table 6: Algorithm-level quantities underlying the pooled analyses. The # HINTs column gives the total number of named recurrent HINTs. The semantic-resampling correlation uses only HINTs satisfying the cross-size source-coverage criterion. A dash indicates that fewer than three eligible paired HINTs remain for the corresponding correlation, or that the correlation is undefined because one of the ranked quantities is constant. Accuracy differences are in percentage points.

Algorithm	# HINTs	$\rho^{R}_{16,64}$	$\rho^{D}_{16,64}$	$A^{\mathrm{ALL}}_{64} - A^{\mathrm{ID}}_{64}$	$\mathrm{mean}\,\lvert I_{64}\rvert - \mathrm{mean}\,\lvert I_{16}\rvert$
activity_selector	3	1.000	1.000	+18.20	+5.23
articulation_points	13	0.508	0.973	+17.78	+4.52
bellman_ford	3	1.000	–	+1.04	+0.00
bfs	2	–	–	+0.00	+0.00
binary_search	3	0.250	0.000	+21.09	+7.12
bridges	12	0.517	0.956	-2.50	+2.99
bubble_sort	3	0.750	–	+19.63	+6.04
dag_shortest_paths	11	0.487	0.800	+2.77	+0.80
dfs	10	0.539	0.886	+20.54	+4.01
dijkstra	5	0.443	0.600	+3.97	+0.89
find_maximum_subarray_kadane	6	0.420	0.231	+16.80	+0.76
floyd_warshall	4	0.800	1.000	+4.01	+1.03
graham_scan	7	0.393	0.971	+25.91	+2.26
heapsort	7	0.107	0.850	+43.74	-0.32
insertion_sort	3	0.250	–	+18.20	+2.83
jarvis_march	6	0.657	0.750	+27.36	+5.24
lcs_length	2	–	–	+8.45	+15.48
matrix_chain_order	3	0.250	–	+4.88	+0.98
minimum	2	–	–	+7.81	+0.00
mst_kruskal	9	0.175	0.900	+34.23	+4.57
mst_prim	5	0.700	0.900	+15.01	+2.48
optimal_bst	4	0.400	0.500	+9.95	+3.91
quickselect	8	0.053	-0.553	+49.48	-4.88
quicksort	5	0.513	0.900	+23.51	+1.05
segments_intersect	5	0.000	–	+0.06	-0.18
strongly_connected_components	12	0.514	0.662	+38.42	+2.39
task_scheduling	3	0.750	0.750	+3.59	+0.23
topological_sort	8	0.598	0.774	+12.92	+5.79

therefore concern structure across the evaluated algorithm cohort and do not require every individual algorithm to exhibit the same intervention effect.