UNDERSTANDING DECISION-MAKING MECHANISMS IN NEURAL ROUTING SOLVERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural Combinatorial Optimization (NCO) has achieved strong empirical success, yet the internal mechanisms driving model decisions remain largely unexplored. In this paper, we investigate three representative autoregressive NCO models spanning two encoder-decoder configurations: AM and POMO (heavy-encoder, light-decoder), and LEHD (light-encoder, heavy-decoder). Through behavioral analyses, representation probing, and causal interventions, we examine how these models construct solutions and use internal representations during decoding. Our results suggest that AM and POMO predominantly follow a persistent geometric pattern throughout solution construction, whereas LEHD contains linearly accessible information about multiple future actions. Causal experiments further provide evidence for the role of future-node representations in LEHD's decision-making. We also observe that LEHD relies strongly on the current-node representation for immediate local decisions, while the start-node representation plays a broader navigational role over the subsequent route. Cross-instance alignment analyses additionally indicate that LEHD maps current-node representations into a relatively shared latent region, which may provide a stable reference for evaluating subsequent decisions. Across the Traveling Salesman Problem and the Capacitated Vehicle Routing Problem, these results reveal distinct decision-making patterns across these architecturally distinct solvers and provide a foundation for more interpretable analyses of NCO solvers. Code and additional visualizations are provided in the supplementary material.

1 INTRODUCTION

The Traveling Salesman Problem (TSP) (Grötschel & Holland, 1991) and the Vehicle Routing Problem (VRP) (Dantzig & Ramser, 1959) are among the most celebrated combinatorial problems in computer science, with critical applications spanning transportation (Pillac et al., 2013), logistics (Konstantakopoulos et al., 2022), and drug discovery (Liu et al., 2017). Due to the NP-hard nature of these routing problems, exact algorithms are computationally prohibitive at scale. While classical heuristics like Concorde (Applegate et al., 2006) and LKH3 (Helsgaun, 2017) produce high-quality solutions, they incur heavy computational overhead. Consequently, Neural Combinatorial Optimization (NCO) has emerged as a significantly faster alternative capable of achieving near-optimal or superior performance on tailored distributions.

Among NCO approaches, Transformer-based architectures (Vaswani et al., 2017) represent a dominant paradigm. Trained via Reinforcement Learning (RL), AM (Kool et al., 2019) employs a heavy encoder and a lightweight decoder (HELD) to construct tours by sequentially appending unvisited nodes to the current partial path. For each instance, AM uses a learned policy to choose the starting node. POMO (Kwon et al., 2021) also uses a HELD architecture and RL to train the model but it also leverages the rotational and structural symmetries inherent to cyclic routing problems and uses different starting positions for each instance, thus not requiring a policy to select a starting node. However, AM and POMO struggle to generalize to problem instances significantly larger than those seen during training. Addressing this bottleneck, LEHD (Luo et al., 2024) shifts to a Supervised Learning (SL) paradigm using an inverted architectural allocation, a light encoder (1 layer) paired with a heavy decoder (6 layers). At each step, LEHD receives the start node, current node, and unvisited set to predict the next step. Beyond autoregressive appending, Luo et al. (2025) recently proposed L2C-Insert, an insertion-based heuristic that decouples generation into node selection and

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.

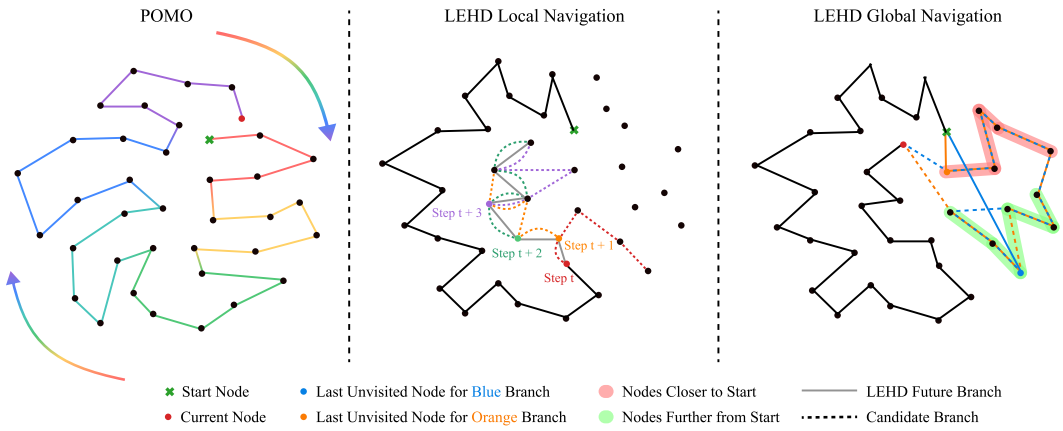


Figure 1: **Comparison of POMO and LEHD tour construction strategies. Left:** POMO constructs the tour in a clockwise manner while oscillating between shallow and deep onion layers, with each oscillation highlighted in a distinct color. **Middle:** LEHD local navigation across consecutive steps. Colored dashed lines represent candidate multi-step plans, which LEHD dynamically revises at each step. **Right:** LEHD global navigation. LEHD evaluates candidate branches based on the relative angular position of the start node relative to the current node. It prioritizes the branch visiting nodes further from the start first (**orange branch**) over the branch visiting closer nodes first (**blue branch**), minimizing the final return cost to the start node.

placement phases, using an encoder-decoder network to determine optimal insertion points within the partial tour.

Despite the rapid evolution of NCO architectures, the internal mechanisms driving their predictions remain largely black boxes. A few pioneering studies have begun peeling back these layers: Zhang et al. (2025) used linear probes to demonstrate that Euclidean distances are linearly decodable from internal representations and identified specific embedding dimensions critical to LEHD's predictions, while Narad et al. (2025) applied Sparse Autoencoders (SAEs) to show that latent neurons capture geometric features such as boundary detection and spatial clustering. While insightful, these prior efforts offer limited mechanistic insights, focusing primarily on correlational analysis rather than establishing causality. Consequently, a fundamental question remains open: *what explicit, macro-level solving strategies do these networks actually learn and execute?* To bridge this gap, we present a comprehensive interpretability framework, combining behavioral probing, representation analysis, activation patching, and causal analysis, to systematically dissect the strategies of leading NCO models. To the best of our knowledge, ours is the first work to employ causal interventions to explain the decision-making mechanisms of these solvers.

Using this framework, we analyze AM, POMO, and LEHD as representative construction paradigms, uncovering distinct spatial and mechanistic behaviors:

- **AM and POMO:** Construct tours through a rigid spatial pattern, sweeping anti-clockwise for AM and clockwise for POMO from the initial node while systematically oscillating between shallow and deep convex (onion) layers (Figure 1). As detailed in Section 4, this structural rigidity stems directly from training on uniform point distributions and shifts predictably when retrained on clustered instances.
- **LEHD:** Exhibits evidence of two complementary decision components which share similarities with Model Predictive Control (MPC) (Richalet et al., 1978). Locally, its final decoder layers encode explicit multi-step trajectories over an effective horizon, executing the immediate step before dynamically revising its trajectory (Figure 1). Globally, LEHD utilizes the relative angular position of the start node to steer navigation toward distant unvisited clusters first, minimizing the eventual return cost. We also find that LEHD maps different current nodes to a relatively shared latent embedding space across instances and steps, suggesting that it plans in a canonical, current-centered latent reference frame.

Our main contributions are summarized as follows:

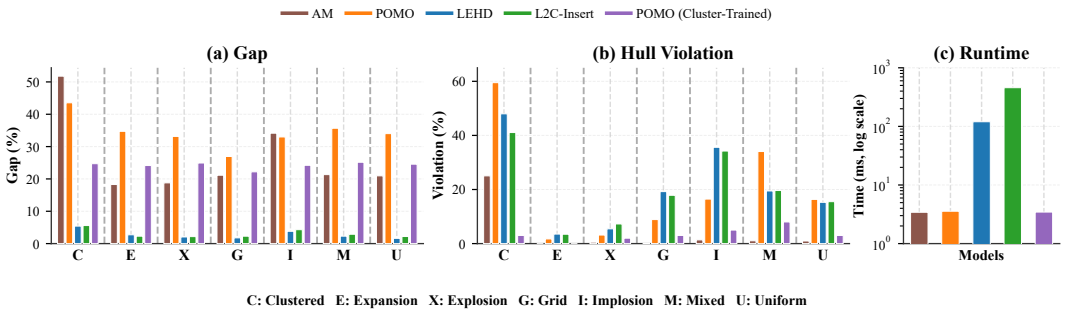


Figure 2: Behavioral comparison of NCO solvers on TSP-500 under seven synthetic distribution shifts. AM, POMO, LEHD, L2C-Insert, and cluster-trained POMO are compared by optimality gap and convex-hull order violations across Clustered, Expansion, Explosion, Grid, Implosion, Mixed, and Uniform distributions, together with inference runtime. The results highlight differences in solution quality, geometric consistency, and computational efficiency under diverse instance geometries.

- We conduct a diverse set of experiments to provide a mechanistic explanation of how distinct NCO architectures construct routing solutions.
- We characterize the geometric structure of the tours constructed by AM and POMO, demonstrating the rigidity of their learned strategy and how the training data distribution influences it.
- We reveal that LEHD coordinates a local look-ahead horizon with global spatial reference-frame navigation, explaining its strong generalization capabilities across problem scales.

Core findings generalize across both TSP and CVRP (details in Appendix F). Appendices A–G cover related work, extended results, extra visualizations, and experimental setups.

2 Behavioral Results and Model Comparison

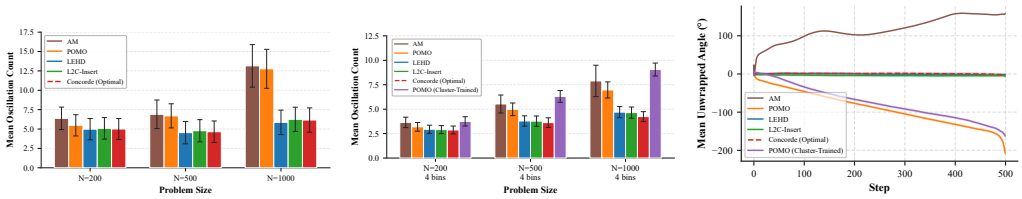
We compare AM, POMO, and LEHD as representative append-based solvers with different encoder–decoder allocations, and include L2C-Insert as an insertion-based reference. Throughout the paper, all models are evaluated using greedy, single-trajectory decoding without model-specific test-time enhancements, such as POMO's rotational augmentation or LEHD's RRC. This evaluation protocol allows us to study the intrinsic decision-making behavior of the learned policies without introducing additional inference-time modifications that may obscure the underlying model mechanisms.

All models are trained on Uniform TSP-100 and evaluated in Figure 2 on TSP-500 across seven distributions, thereby testing both size and distribution generalization. We report optimality gap, convex-hull violation rate, and inference runtime on a logarithmic scale; a hull violation denotes failure to preserve the cyclic order of convex-hull vertices, a necessary property of an optimal Euclidean TSP tour.

L2C-Insert incurs substantially higher inference costs without consistently improving solution quality over LEHD. LEHD achieves competitive solution quality across diverse distributions while maintaining substantially lower inference cost than the insertion-based baseline. Therefore, we retain L2C-Insert as an insertion-based reference for behavioral comparison, while focusing our mechanistic analyses on AM, POMO, and LEHD due to their shared append-based construction process, which enables a controlled comparison of their decision mechanisms. Additional details are provided in Appendix C.

3 Experimental Framework

This section presents the experimental framework used to interpret NCO solvers. Following the problem formulations in Appendix A, we first characterize their observable geometric behavior and then examine the representational and causal mechanisms underlying their decisions. The experimental settings are provided in Appendix G and the corresponding CVRP analyses are reported in



(a) Mean onion layer oscillation (b) Mean intra-cluster oscillation (c) Angle comparison per step

Figure 3: **Tracking geometrical properties of generated tours.** **(a)** Onion layers are segmented into B bins to calculate the average number of oscillations between shallow and deeper layers, where higher values indicate a greater number of deep oscillations. **(b)** Each cluster is partitioned into binwise onion layers to count high-intensity oscillations within individual clusters. **(c)** The average start-to-current node angle relative to each tour's first edge is tracked throughout tour construction. A decreasing angle indicates a clockwise rotation of the start-to-current displacement vector.

Appendix F. These analyses are subsequently synthesized in Section 4 and Section 5 to develop an interpretation of the distinct decision-making strategies learned by AM, POMO, and LEHD.

3.1 Geometric Trajectory Analysis

We characterize the generated solutions through two complementary geometric summaries: onion-depth progression and start-centered angular progression. The first measures how a solver moves across nested convex layers of the point set, while the second describes how selected nodes are distributed angularly with respect to the first generated edge.

3.1.1 Onion-Depth Progression

We use onion decomposition to characterize the layer-wise structure of TSP instances (Chazelle, 1985; de Berg et al., 2000). It recursively removes the convex hull of the remaining points, partitioning them into nested layers ρ_t (Appendix A.3). We then discretize $\rho_t \in [0, 1]$ into B equal-width bins $b_t = 1 + \min(B - 1, \lfloor B\rho_t \rfloor)$.

Bin 1 is the outermost region and bin B the deepest. After removing consecutive duplicate bins, we count an oscillation when the trajectory leaves bin 1, reaches at least $b_{\min} = \min(B, \lfloor \tau B \rfloor + 1)$, and returns to bin 1, where $\tau \in (0, 1]$ controls the required depth. We report the average number of such oscillations across problem instances.

3.1.2 Angular Progression

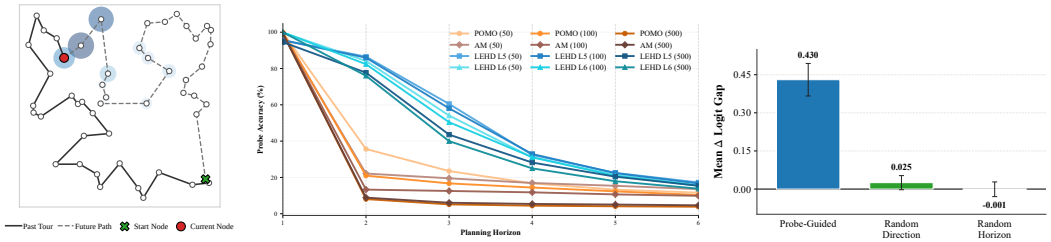
We measure start-centered angular progression using the first generated edge as the reference axis, $a = x_{\pi_1} - x_{\pi_0}$, where π_0 is the start node. At step $t \in \{1, \ldots, N-1\}$, we compare it with $v_t = x_{\pi_t} - x_{\pi_0}$. The signed angular position is computed as $\theta_t = \operatorname{atan2}\left(a^x v_t^y - a^y v_t^x, a^\top v_t\right)$.

We unwrap θ_t to remove artificial discontinuities at the $[-180°, 180°]$ boundary, allowing the angular progression to vary continuously beyond this interval. We then compute the mean signed angle across instances at each decoding step, capturing systematic clockwise or counterclockwise progression around the start node.

For CVRP, angles are measured within each route relative to its first edge, excluding depot returns. Routes are aligned by normalized progress before aggregation.

3.2 Future-Action Planning Probes

We use linear probes to examine whether neural combinatorial solvers encode information about their own future decisions. At decoding step t, the frozen model has produced the partial solution $\pi_{\leq t} = [\pi_0, \pi_1, \ldots, \pi_t]$, and the current decoding state defines a feasible candidate set $\mathcal{A}_t$. For TSP, this set consists of the unvisited nodes, $\mathcal{A}_t = V \setminus \{\pi_0, \ldots, \pi_t\}$. For routing problems with additional constraints, such as CVRP, $\mathcal{A}_t$ contains all unvisited customers, regardless of current



(a) LEHD attention (b) Probe accuracy per planning horizon (c) Probe-guided steering

Figure 4: **(a)** LEHD attention heatmap from the current node to the unvisited nodes. **(b)** Probe accuracy across planning horizons for the decoder layers of LEHD, AM, and POMO. **(c)** Probe-guided counterfactual steering for LEHD on TSP-100 at $\alpha = 5$, compared against random-direction and random-horizon baselines. Bars show the mean change in logit gap between the clean top-two actions

capacity constraints. For a future horizon h, the probing target is the action selected by the same frozen model at step $t + h$ during its greedy rollout, $y_{t,h} = \pi_{t+h}$. The probe therefore does not predict the optimal solution; instead, it tests whether the model's current internal representation already contains information about its own future trajectory. For each feasible candidate $i \in \mathcal{A}_t$, we extract a 128-dimensional candidate-specific representation $r_{t,i}^{(\ell)}$ from internal layer ℓ. A separate linear probe is trained for each layer ℓ and horizon h, assigning one scalar score to every feasible candidate: $s_{t,i}^{(\ell,h)} = W_{\ell,h} r_{t,i}^{(\ell)} + b_{\ell,h}$. The probe is optimized using cross-entropy loss.

This formulation treats future-action prediction as a candidate-ranking problem. If a linear probe can recover the model's future action π_{t+h} from the representation at step t, then that representation contains linearly accessible information about the model's subsequent decisions. We apply the same probing protocol to the POMO, AM, and LEHD (Figure 4b), while the model-specific representation choices are described in their corresponding analysis sections.

3.3 Probe-Guided Counterfactual Steering

Inspired by prior work combining planning probes with causal interventions and activation steering (Bush et al., 2025; Li et al., 2023; Panickssery et al., 2023; Li et al., 2022; Nanda et al., 2023), we investigate whether amplifying representations associated with an alternative future route can shift the model's immediate action preference.

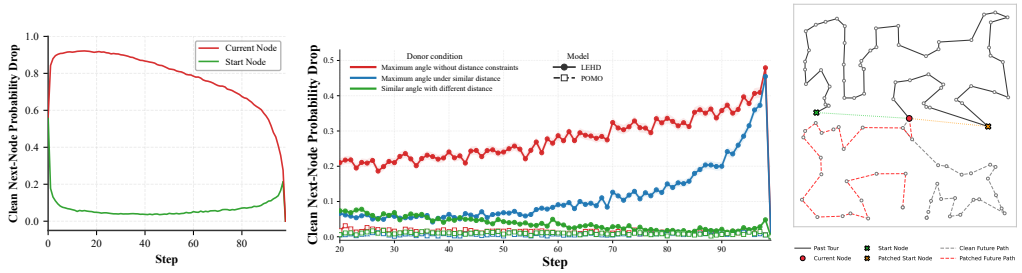
At decoding step t, let a and a' denote the highest- and second-highest-probability feasible actions, respectively. We construct a counterfactual continuation by forcing $\pi'_{t+1} = a'$ and then following the frozen model's greedy policy. We ask whether intervening on representations of nodes appearing at future steps along this counterfactual continuation can causally shift the model's immediate action preference toward a'.

For future horizons $h \in \mathcal{H}$ and decoder layers $\ell \in \mathcal{L}$, we steer the representation of the counterfactual future node $y'_{t,h} = \pi'_{t+h}$ along its probe direction: $\tilde{r}_{t,y'_{t,h}}^{(\ell)} = r_{t,y'_{t,h}}^{(\ell)} + \alpha W_{\ell,h}^{\top} / \|W_{\ell,h}\|_2^2$, where $\alpha > 0$ controls intervention strength. We jointly steer horizons $\mathcal{H} = \{2, 3, 4, 5\}$, without directly modifying the immediate candidates a or a'.

We then recompute the action logits without forcing a' and measure the change in its preference over a: $\Delta G_t = [z_{\text{steered}}(a') - z_{\text{steered}}(a)] - [z_{\text{clean}}(a') - z_{\text{clean}}(a)]$. Controls use norm-matched random directions or an equal number of disjoint random future horizons.

3.4 Node-Role Representation Alignment

We measure the cross-instance cosine similarity of node representations at each decoding step. For each layer, decoding step, and node role, we compare the corresponding representations across problem instances. This quantifies how consistently each functional role is represented across instances throughout decoding.



(a) LEHD mean ablation (b) POMO vs. LEHD start-node patching (c) LEHD trajectory

Figure 5: Causal interventions on node representations. **(a)** Drop in the clean next-node probability after mean ablation of the current and start nodes. **(b)** Effect of start-node activation patching under three geometric donor conditions for LEHD and POMO. **(c)** Example clean and patched LEHD trajectories showing route changes over 22 decoding steps.

3.5 Current and Start Node Contribution

We use activation patching (Appendix B.2) to examine the roles of two state-defining node representations during decoding: the current node and the start node. The current node represents the local position from which the next decision is made, while the start node is a fixed element of the generated tour that remains available to the decoder throughout the rollout. In both interventions below, we modify the encoder-produced node embeddings before they are passed to the decoder.

3.5.1 Node Attribution

We perform stepwise mean ablations of the start and current node representations. At each decoding step, we compute a mean embedding over all clean encoded node representations collected from a reference set of instances at that step, and temporarily replace either the start-node or current-node representation with this mean. This removes instance-specific information while keeping the remaining decoding state unchanged. We then measure the drop in the clean next-node probability to assess the influence of each representation on the immediate decision.

3.5.2 Targeted Intervention

In the start-node patching experiment, we replace the encoded representation of the original start node with the representation of an already visited donor node, i.e., $H'_{\pi_0} = H_d$, where π_0 is the original start node and d is the donor node. This intervention changes the start-node representation while leaving the decoding state otherwise unchanged. We select donor nodes using three geometric conditions: maximum angle without distance constraints, maximum angle under similar distance, and similar angle with different distance. These conditions allow us to separate the effects of angular direction and distance in the start-node representation.

4 Interpretability Analysis of AM and POMO

As noted in Section 1, both AM and POMO employ an asymmetrical architecture consisting of multiple encoder layers and a single decoder layer. Because the current decoding step is omitted from the encoder embeddings and processed exclusively by the lightweight decoder, we hypothesize that AM and POMO's construction strategy relies less on local geometric structures around the current node and more on a generalized global heuristic. To test this hypothesis, we examine the geometric properties of tours generated by these models and compare them against alternative solvers.

Following the methodology in Section 3.1, we track the angular displacement of the current node relative to the start node throughout tour construction. As shown in Figure 3c, a clear behavioral divergence emerges: while baseline methods dynamically adjust their angular direction without an explicit directional bias, **POMO exhibits a systematic tendency to construct tours in a clockwise manner while AM constructs in an anti-clockwise manner**.

Furthermore, **both of these models oscillate between different onion-layer depths with substantially higher frequency and intensity** than competing solvers (Figure 3a). This spatial disparity becomes increasingly pronounced on larger problem scales, helping explain why AM and POMO underperform relative to LEHD and L2C-Insert, both of which adapt more dynamically to local instance geometry (see Section 5).

This observation aligns with findings by Huang et al. (2025), who noted that lightweight decoder architectures yield less context-aware decoding decisions, ultimately limiting performance. To mitigate this constraint, light-decoder architectures frequently employ multi-start sampling at inference time, generating candidate tours from multiple initial nodes per instance and selecting the minimal-cost path. We argue that the efficacy of multi-start sampling stems directly from counteracting POMO's rigid geometric strategy. However, because test-time sampling provides only constrained flexibility compared to the innate adaptivity of greedy LEHD, the performance gap between the two models widens as instance sizes increase (Appendix C).

This structural rigidity also accounts for AM and POMO's performance degradation on clustered distributions (Figure 2). Because these models are trained exclusively on uniform point distributions, their learned heuristics function adequately on uniform data but fail to generalize to clustered instances, forcing unnecessary inter-cluster jumps that severely inflate total tour cost and hull violation. Interestingly, while POMO suffers a larger overall performance gap, its rigid clockwise progression better preserves the cyclic order of convex-hull nodes on near-uniform distributions, while degrading sharply on clustered and mixed distributions (Figure 2). This also applies to AM.

To evaluate how training data distributions influence POMO's learned strategy, we retrained the model on clustered datasets from Bi et al. (2023) and measured intra-cluster layer oscillations. As illustrated in Figure 3b, this retrained variant oscillates significantly more within individual clusters than baseline methods—and even exceeds the oscillation intensity of the original uniform-trained POMO. This shift demonstrates that the training distribution directly governs the learned heuristic: when trained on clustered instances, POMO restricts its layer oscillations within a cluster before transitioning to the next. This adaptation yields substantial performance gains across all evaluated instance distributions (Figure 2), suggesting that diversifying training data distributions offers a promising avenue for enhancing NCO generalization more broadly. We trained AM and LEHD on clustered data as well, as depicted in Appendix C.1.

Remarkably, the angular progressions of AM and POMO persist even in their cluster-trained variant, indicating that they apply their core strategy hierarchically within each cluster, effectively treating individual clusters as distinct subproblems. This finding reinforces the insights of Bi et al. (2023), demonstrating that light-decoder models operate similarly when solving full instances or decomposed subproblems.

5 Interpretability Analysis of LEHD

While POMO's routing strategy is relatively rigid, as discussed in Section 4, LEHD's strategy is considerably more adaptive to the underlying geometry of the nodes. Our analysis reveals that LEHD's navigation paradigm comprises two distinct components: a *local navigation* mechanism that optimizes paths across immediate local neighborhoods, and a *global navigation* mechanism that steers the local trajectory toward a broader, global target direction. Below, we dissect each component in detail.

5.1 Local Navigation

How does LEHD select its immediate next node—and does it plan ahead when doing so? Because LEHD computes attention over all nodes during decoding, we first visualized its attention heatmaps. This revealed a striking pattern (see Figure 4a): at any given step, the current node attends heavily not only to the immediate next node but also to a sequence of nodes slated for addition in subsequent decoding steps. This suggests that LEHD selects its next node by explicitly accounting for its planned trajectory over a longer horizon, a strategy that shows some resemblance to Model Predictive Control (MPC) (Richalet et al., 1978). In MPC, a future path is optimized up to a fixed predictive horizon, but only the first control step is executed; looking ahead prevents short-sighted, purely greedy decisions.

To test whether future trajectory information is actively encoded in the decoder, we trained linear classifier probes on the latent representations from the final two layers. The full probing protocol is detailed in Section 3.2. We define the horizon as the number of decoding steps between the current node and the node being predicted.

Probing accuracies are shown in Figure 4b, alongside AM and POMO as baselines. For POMO and AM, we probe candidate-wise products of decoder contexts and final encoder embeddings or logit keys, respectively. At horizon 1 (the immediate next node), all models achieve near-perfect accuracy, as expected: all three architectures use a linear projection over these candidate-specific representations to compute the next-node distribution. At longer horizons, however, a stark divergence emerges. LEHD predicts the node two steps ahead with approximately 80% accuracy, indicating that substantial future-path information is explicitly retained in its latent space. AM and POMO, by contrast, degrade sharply, especially on larger instances, implying that such information is absent or heavily obscured in their representations. This gap persists up to horizons of 4–5 steps, beyond which LEHD's probe accuracy also decays, delineating the upper bound of its effective look-ahead horizon. To rule out the possibility of the probes predicting the nearest neighbor, in Section D.1 we also evaluate them on instances where LEHD does not choose the nearest neighbor.

Probing establishes the *presence* of future-path information, but not that the model uses it. To test causal relevance, we turn to the counterfactual steering described in Section 3.3. As shown in Figure 4c, slightly modifying the embeddings of nodes in the second most probable branch changes the model's preference toward choosing a'_1 as its immediate next action, without altering the embeddings of the immediate next nodes. Random-direction updates, or updates to nodes far along the horizon, leave LEHD's decision unchanged, consistent with the bounded effective horizon identified by the probes. We further assess potential off-manifold effects of these interventions in Appendix D.3. This provides causal evidence that LEHD relies on its encoded look-ahead trajectory to determine its local actions. A complementary behavioral causal experiment in Appendix D.2 further examines POMO and LEHD and corroborates this conclusion.

Table 1: TSP-100 rollout-order similarity comparison between LCS and Rev. LCS.

Angle Range	LEHD		POMO	
	LCS	Rev. LCS	LCS	Rev. LCS
0°–30°	**0.769**	0.147	**0.963**	0.030
30°–60°	**0.684**	0.226	**0.963**	0.033
60°–90°	**0.612**	0.302	**0.965**	0.036
90°–120°	**0.516**	0.398	**0.966**	0.039
120°–150°	0.428	**0.489**	**0.963**	0.042
150°–180°	0.391	**0.544**	**0.963**	0.052

Table 2: Layer-wise cross-instance alignment of current, start, and random-node representations in LEHD on Uniform TSP-100. Enc. and Dec. denote encoder and decoder layers, respectively.

Layer	Current	Start	Random
Enc. 1	0.194 ± 0.557	0.188 ± 0.559	0.193 ± 0.557
Dec. 1	0.378 ± 0.435	$\mathbf{0.457} \pm 0.305$	0.437 ± 0.336
Dec. 2	$\mathbf{0.767} \pm 0.124$	0.595 ± 0.256	0.378 ± 0.360
Dec. 3	$\mathbf{0.741} \pm 0.114$	0.630 ± 0.222	0.371 ± 0.339
Dec. 4	$\mathbf{0.886} \pm 0.046$	0.715 ± 0.175	0.375 ± 0.306
Dec. 5	$\mathbf{0.732} \pm 0.109$	0.691 ± 0.167	0.373 ± 0.269

5.2 Global Navigation

While local navigation governs immediate look-ahead dependencies, LEHD also relies on a global navigation component that steers the route toward a broader target direction. To isolate the role of the start node in this mechanism, we first compare it with the current node using the mean-ablation intervention described in Section 3.5.1. As shown in Figure 5a, ablating the current node produces a much larger drop in the clean next-node probability than ablating the start node. This indicates that immediate next-node selection is governed primarily by the current local state, whereas the start node has a weaker direct effect on local transitions and may instead support higher-level guidance.

We examine this role more directly using the activation-transfer intervention introduced in Section 3.5.2, where the original start-node representation S is replaced by that of a donor node S'. As shown in Figure 5b, *maximum angle without distance constraints* produces the largest drop in the clean next-node probability, followed by *maximum angle under similar distance*, while *similar angle with different distance* produces the smallest effect. The first two conditions preserve a large angular displacement under different distance constraints, whereas the third substantially changes distance while approximately preserving direction. This ordering suggests that the intervention effect is more

strongly associated with the angular displacement $\angle(S, C, S')$, defined by the original start node, the current node, and the donor node, than with the distance between S and S'. In contrast, the corresponding POMO curves remain comparatively close to zero and exhibit little separation across the three donor conditions. Thus, POMO's immediate decisions appear largely insensitive to the directional information introduced through the patched start representation, whereas LEHD responds strongly to changes in this direction. The corresponding results for AM, which selects its starting node through its learned policy, are discussed in Section D.4.

The qualitative example in Figure 5c illustrates the resulting reorientation. In this instance, the patched start node lies at an angle of nearly 180° relative to the original start node, and the generated route changes its overall direction so that the remaining trajectory approaches the patched anchor through a shorter completion path. To quantify this effect, Table 1 compares the ordering of the unvisited nodes in the clean and patched rollouts. LCS is the normalized longest common subsequence between the clean rollout order and the patched rollout order, and therefore measures how much of the original forward ordering is preserved. Rev. LCS instead compares the patched rollout order with the reversed clean rollout, and measures whether the intervention causes the remaining nodes to be visited in the opposite direction.

For LEHD, increasing $\angle(S, C, S')$ consistently decreases LCS and increases Rev. LCS. At the largest angular displacements, reversed agreement exceeds forward agreement, indicating a substantial reordering of the future selection sequence toward the reverse direction. In contrast, POMO maintains relatively high LCS and low Rev. LCS across the full range of angular interventions, suggesting that its rollout order remains comparatively stable even under large directional perturbations. Together, these results suggest that the start node provides global directional guidance in LEHD, but has much less influence in POMO.

5.3 Latent Space

To examine whether LEHD organizes node representations according to their functional roles, we apply the cross-instance alignment analysis described in Section 3.4. As shown in Table 2, the current-node representation exhibits high cosine similarity across instances from decoder layer 2 onward. Although the current node changes at every decoding step and occupies different geometric locations across instances, its representation remains strongly aligned at the same step. This suggests that LEHD maps functionally equivalent current nodes into a shared latent region, potentially providing a consistent reference for subsequent decisions. The start node, by contrast, remains fixed throughout the rollout, yet its alignment increases more gradually and peaks in deeper decoder layers, consistent with a broader navigational role becoming more strongly expressed later in the computation. Randomly selected unvisited nodes show larger variability and less consistent alignment. Overall, these results suggest that LEHD progressively organizes its latent space around distinct task-specific node roles. Additional visualizations are provided in Appendix E.

6 Conclusion and Future Work

In this work, we conducted an extensive and diverse set of experiments to provide a comprehensive understanding of the strategies underlying three of the most well-known neural NCO solvers for routing problems. Our results reveal that these models employ distinct strategies to solve TSP and CVRP instances. These insights, and the methods used to derive them, can inform the design of better solvers and guide future work aimed at deepening our understanding of these models.

Several promising future directions emerge from this study. First, training on more diverse data distributions could improve robustness and generalization. Second, LEHD could be trained to predict multiple nodes along the path at once instead of only the next node, enabling it to plan further ahead and potentially strengthening its routing capability. Third, increasing the number of decoder layers relative to encoder layers appears to increase model flexibility and improve generalization. While this work focuses on three representative NCO solvers, extending the analysis to other solvers is another interesting direction. Extended results and visualizations are available in the supplementary material.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods (specifically, assisting in the implementation of some experiments). We have not used generative AI tools for generating synthetic data sets, developing theoretical models or conceptual frameworks, formulating or proving mathematical claims, proposing or refining hypotheses, designing research methodology or experiments, supporting qualitative or thematic data analysis, interpreting results, or assisting with translation or dataset cleaning and reformatting. Additionally, we used generative AI tools for creating or editing software code (primarily for initial code scaffolding, such as generating plots or loading raw data) and for editing the paper to improve readability (proofreading and polishing the text of this manuscript). We have reviewed all AI-assisted work. Specifically, all code generated with LLM assistance was rigorously reviewed, tested, and validated by multiple co-authors to ensure its correctness and integration into our codebase, and all AI-assisted text edits were reviewed by the authors for accuracy and intent. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2018. URL https://arxiv.org/abs/1610.01644. 14

David L. Applegate, Robert E. Bixby, Vasek Chvátal, and William J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006. 1

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances, 2021. URL https://arxiv.org/abs/2102.12452. 15

Jieyi Bi, Yining Ma, Jiahai Wang, Zhiguang Cao, Jinbiao Chen, Yuan Sun, and Yeow Meng Chee. Learning generalizable models for vehicle routing problems via knowledge distillation, 2023. URL https://arxiv.org/abs/2210.07686. 7

Thomas Bush, Stephen Chung, Usman Anwar, Adrià Garriga-Alonso, and David Krueger. Interpreting emergent planning in model-free reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 82115–82197, 2025. 5

B. Chazelle. On the convex layers of a planar set. *IEEE Transactions on Information Theory*, 31(4): 509–517, 1985. doi: 10.1109/TIT.1985.1057060. 4

G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Science*, 6:80–91, 10 1959. 1

Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, second edition, 2000. URL http://www.cs.uu.nl/geobook/. 4

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, et al. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Journal of Machine Learning Research*, 26(83):1–64, 2025. 15

Martin Grötschel and Olaf Holland. Solution of large-scale symmetric travelling salesman problems. *Math. Program.*, 51(1–3):141–202, July 1991. ISSN 0025-5610. 1

Stefan Heimersheim and Neel Nanda. How to use and interpret activation patching, 2024. URL https://arxiv.org/abs/2404.15255. 15

Keld Helsgaun. *An Extension of the Lin-Kernighan-Helsgaun TSP Solver for Constrained Traveling Salesman and Vehicle Routing Problems: Technical report*. Roskilde Universitet, December 2017. 1

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8): 1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735. 14

Ziwei Huang, Jianan Zhou, Zhiguang Cao, and Yixin XU. Rethinking light decoder-based solvers for vehicle routing problems. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4pRwkYpa2u. 7

Erik Jenner, Shreyas Kapur, Vasil Georgiev, Cameron Allen, Scott Emmons, and Stuart Russell. Evidence of learned look-ahead in a chess-playing neural network. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385. 15

Daisuke Kikuta, Hiroki Ikeuchi, Kengo Tajiri, and Yuusuke Nakano. *RouteExplainer: An Explanation Framework for Vehicle Routing Problem*, pp. 30–42. Springer Nature Singapore, 2024. ISBN 9789819722594. doi: 10.1007/978-981-97-2259-4_3. URL http://dx.doi.org/10.1007/978-981-97-2259-4_3. 15, 16

Grigorios D. Konstantakopoulos, Sotiris P. Gayialis, and Evripidis P. Kechagias. Vehicle routing problem and related algorithms for logistics distribution: a literature review and classification. *Operational Research*, 22(3):2033–2062, July 2022. doi: 10.1007/s12351-020-00600-7. URL https://ideas.repec.org/a/spr/operea/v22y2022i3d10.1007_s12351-020-00600-7.html. 1

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems!, 2019. URL https://arxiv.org/abs/1803.08475. 1, 14

Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. Pomo: Policy optimization with multiple optima for reinforcement learning, 2021. URL https://arxiv.org/abs/2010.16011. 1, 14

Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. *arXiv preprint arXiv:2210.13382*, 2022. 5

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in neural information processing systems*, 36:41451–41530, 2023. 5

Ruiwu Liu, Xiaocen Li, and Kit S Lam. Combinatorial chemistry in drug discovery. *Current Opinion in Chemical Biology*, 38:117–126, 2017. ISSN 1367-5931. doi: https://doi.org/10.1016/j.cbpa.2017.03.017. URL https://www.sciencedirect.com/science/article/pii/S1367593117300534. Next Generation Therapeutics. 1

Fu Luo, Xi Lin, Fei Liu, Qingfu Zhang, and Zhenkun Wang. Neural combinatorial optimization with heavy decoder: Toward large scale generalization, 2024. URL https://arxiv.org/abs/2310.07985. 1, 14

Fu Luo, Xi Lin, Mengyuan Zhong, Fei Liu, Zhenkun Wang, Jianyong Sun, and Qingfu Zhang. Learning to insert for constructive neural vehicle routing solver, 2025. URL https://arxiv.org/abs/2505.13904. 1, 14

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. In *Advances in neural information processing systems*, 2022. 15

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. *arXiv preprint arXiv:2309.00941*, 2023. URL https://arxiv.org/abs/2309.00941. 5

Reuben Narad, Leonard Boussioux, and Michael Wagner. Mechanistic interpretability for neural tsp solvers, 2025. URL https://arxiv.org/abs/2510.21693. 2, 15, 16

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition, 2024. *URL https://arxiv.org/abs/2312.06681*, 3, 2023. 5

Victor Pillac, Michel Gendreau, Christelle Guéret, and Andrés L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, 2013. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2012.08.015. URL https://www.sciencedirect.com/science/article/pii/S0377221712006388. 1

J. Richalet, A. Rault, J.L. Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14(5):413–428, 1978. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(78)90001-8. URL https://www.sciencedirect.com/science/article/pii/0005109878900018. 2, 7

Mohammad Taufeeque, Philip Quirke, Maximilian Li, Chris Cundy, Aaron David Tucker, Adam Gleave, and Adrià Garriga-Alonso. Planning in a recurrent neural network that plays sokoban, 2025. URL https://arxiv.org/abs/2407.15421. 15

Mohammad Taufeeque, Aaron David Tucker, Adam Gleave, and Adrià Garriga-Alonso. Path channels and plan extension kernels: a mechanistic description of planning in a sokoban RNN. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=aAshH4kQ1v. 15

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf. 1

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks, 2017. URL https://arxiv.org/abs/1506.03134. 14, 15

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022. 15

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992. 14

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. *arXiv preprint arXiv:2309.16042*, 2023. 15

Zhiqin Zhang, Yining Ma, Zhiguang Cao, and Hoong Chuin Lau. Probing neural combinatorial optimization models, 2025. 2, 15, 16

A PRELIMINARIES

A.1 TRAVELING SALESMAN PROBLEM

Given a complete graph $G = (V, E)$ with n nodes and edge costs c_{ij}, the TSP seeks a minimum-cost Hamiltonian cycle visiting each node exactly once. Formally, we seek a permutation $\pi = [\pi_0, \pi_1, \ldots, \pi_{n-1}]$ that minimizes:

$$\min_{\pi} \left(\sum_{i=0}^{n-2} c_{\pi_i, \pi_{i+1}} + c_{\pi_{n-1}, \pi_0} \right). \tag{1}$$

A.2 CAPACITATED VEHICLE ROUTING PROBLEM

In the CVRP, a vehicle with capacity Q must service customer demands $d_i > 0$ from a central depot v_0. The goal is to find a set of routes $\mathcal{R}$ minimizing total travel cost while respecting capacity constraints:

$$\begin{aligned} \min_{\mathcal{R}} \quad & \sum_{\mathcal{R}_k \in \mathcal{R}} \sum_{t=0}^{|\mathcal{R}_k|-2} c_{r_t^k, r_{t+1}^k}, \\ \text{s.t.} \quad & \sum_{t=1}^{|\mathcal{R}_k|-2} d_{r_t^k} \leq Q, \quad \forall \mathcal{R}_k \in \mathcal{R}. \end{aligned} \tag{2}$$

Each route $\mathcal{R}_k = [r_0^k, r_1^k, \ldots, r_{|\mathcal{R}_k|-1}^k]$ starts and ends at the depot ($r_0^k = r_{|\mathcal{R}_k|-1}^k = v_0$), and every customer is visited exactly once.

A.3 ONION DECOMPOSITION

Onion decomposition recursively removes the convex hull of the remaining points, partitioning them into nested layers. A node has onion depth $d(v) = k$ if it is removed at peeling iteration k; outer-hull nodes have depth zero, while larger values indicate deeper layers.

Let π_t denote the node selected at decoding step t. We normalize its depth as

$$\rho_t = \frac{d(\pi_t)}{\max(1, \max_{v \in \mathcal{V}} d(v))}. \tag{3}$$

B Related Work

B.1 NCO Solvers

B.1.1 Pointer Networks

The paradigm of using deep neural networks to learn heuristics for solving NP-hard routing problems was pioneered by Pointer Networks (Vinyals et al., 2017). Unlike traditional sequence-to-sequence models that are constrained by a fixed output vocabulary, Pointer Networks employ a modified attention mechanism as a dynamic pointing layer, enabling the model to select outputs directly from variable-length input sequences.

B.1.2 AM

While Pointer Networks were originally implemented using recurrent architectures such as LSTMs (Hochreiter & Schmidhuber, 1997), the Attention Model (AM) (Kool et al., 2019) adopted a Transformer-based architecture. AM achieves permutation invariance, allowing the model to treat the input graph as an unordered spatial set. To eliminate the need for ground-truth optimal solutions during training, AM introduced the use of reinforcement learning through the REINFORCE algorithm (Williams, 1992), together with a rollout baseline to reduce gradient variance.

B.1.3 POMO

Although AM demonstrated the effectiveness of Transformer-based routing solvers, its training process still suffered from high variance. Policy Optimization with Multiple Optima (POMO) (Kwon et al., 2021) addressed this limitation by exploiting the rotational and structural symmetries inherent in cyclic routing problems such as the TSP. Instead of constructing a single trajectory, POMO generates N trajectories for a given graph instance, each initialized from a different starting node.

B.1.4 LEHD

A persistent limitation of both AM and POMO is their difficulty in generalizing to graph sizes substantially larger than those encountered during training. Unlike POMO, LEHD (Luo et al., 2024) adopts a supervised learning paradigm. It employs a Transformer-based architecture consisting of a one-layer encoder and a six-layer decoder. At each decoding step, the model receives the starting node, the current node, and the set of unvisited nodes, and predicts the next node to append to the partial route.

B.1.5 L2C-Insert

Both POMO and LEHD are construction heuristics that generate solutions through successive node appending. Aiming to overcome the limitations of appending-based strategies, Luo et al. (2025) proposed L2C-Insert, an insertion-based solver. The method consists of a node-selection phase, in which a candidate node is chosen based on its proximity to the previously selected node, followed by an insertion phase. During the insertion phase, a Transformer encoder-decoder architecture determines the optimal position of the selected node within the current partial tour, thereby incrementally expanding the solution.

B.2 Mechanistic Interpretability

Mechanistic Interpretability (MI) seeks to uncover the internal computations performed by neural networks by identifying the information encoded in their representations and establishing causal relationships between model components and observed behavior.

B.2.1 Probing

Probing is one of the most widely used approaches for analyzing neural representations. A probe is typically a simple linear classifier or regressor trained to predict a target property from a model's hidden activations. Strong probe performance suggests that information relevant to the target property is encoded in the examined representation (Alain & Bengio, 2018). Consequently, probing has

become a standard tool for investigating the information accessible at different layers of a neural network (Belinkov, 2021).

B.2.2 Activation Patching

While probing can reveal whether information is present in a representation, it cannot determine whether that information is causally used by the model. Activation patching addresses this limitation by replacing activations from one input with the corresponding activations from another and measuring the resulting change in model behavior. By analyzing the effects of such interventions, activation patching can identify components that are causally responsible for specific computations (Meng et al., 2022; Wang et al., 2022; Zhang & Nanda, 2023; Geiger et al., 2025). The technique has been successfully applied to uncover computational circuits and information flow in Transformer-based models (Heimersheim & Nanda, 2024).

B.2.3 Look-Ahead and Planning in Sequential Decision-Making Models

Outside of NLP, a closely related line of MI work asks whether networks trained end-to-end on sequential decision-making tasks represent their own future actions, and whether these representations are causally used rather than merely correlated with behavior. Jenner et al. (2024) combine linear probing with activation patching on Leela Chess Zero, the strongest open-source chess engine, and find that its policy network linearly encodes the optimal move several turns ahead of the current position, and that these representations are causally necessary for its output in certain board states. In a parallel line of work on a different planning domain, Taufeeque et al. (2025) train linear probes that decode a recurrent network's future actions roughly 50 steps in advance while it plays Sokoban, and show via intervention on the hidden state that these probed representations causally steer the agent's subsequent behavior; Taufeeque et al. (2026) extend this analysis into a full circuit-level account, localizing directional "path channels" that implement a bidirectional, plan-extending search. Both lines of work establish the same two-step recipe that we adopt: probing to establish that a representation of a future decision exists, followed by causal intervention to establish that the model actually relies on it. Our contribution is to bring this recipe to bear on NCO routing solvers, where a variable-size, permutation-sensitive candidate set and the absence of a fixed board or grid structure require a different probe formulation — candidate-ranking over the feasible action set rather than a fixed square- or move-indexed classification target — than either the chess or Sokoban settings.

B.3 Interpretability of NCO Solvers

Despite the rapid progress of neural combinatorial optimization, the internal mechanisms underlying these solvers remain largely unexplored. One of the first studies to investigate this question is the work of Zhang et al. (2025). Using probing techniques, they showed that Euclidean distance information is encoded in the internal representations of AM, POMO, and LEHD. Furthermore, their analysis suggested that these models do not behave purely myopically when constructing solutions: their myopia-avoidance probe is framed as a single-step, binary classification between the globally optimal edge and the locally greedy (nearest-neighbor) edge at the current decision, evaluated on the same two backbones we study. This probe does not test whether the model encodes a specific candidate action at longer horizons, nor whether any such representation is causally used, both of which are the focus of our Future-Action Planning Probes and the causal interventions. Also by examining the weights of the probe they found that there exists two dimensions in the latent space of LEHD which have high importance in its node selection.

More recently, Narad et al. (2025) applied sparse autoencoders (SAEs) to a pointer network-style model (Vinyals et al., 2017) and found evidence that the learned representations capture interpretable geometric concepts, including boundary detection and spatial clustering; the authors explicitly identify causal circuit analysis via activation patching as future work rather than something their study performs.

Complementary to these mechanistic approaches, Kikuta et al. (2024) proposed RouteExplainer, a post-hoc explanation framework for VRP solutions. RouteExplainer quantifies the influence of individual edges on the generated route, introduces a pipeline for generating counterfactual explanations, and leverages large language models (LLMs) to enhance the interpretability of the resulting explanations.

Taken together, existing interpretability studies of NCO solvers are exclusively correlational — probing (Zhang et al., 2025), sparse autoencoders (Narad et al., 2025), post-hoc explanation (Kikuta et al., 2024), — and none establishes a causal link between a specific internal representation and a specific routing decision. To the best of our knowledge, the activation-patching experiments in Section 4.4 are the first causal account of decision-making mechanisms in NCO routing solvers specifically, extending the look-ahead-probing-and-patching paradigm established outside NLP (above) to this domain for the first time.

C Extended Behavioral Results

This section extends the behavioral evaluation presented in the main paper. We report detailed results for TSP and CVRP across multiple problem sizes and seven distribution shifts, and subsequently examine the rotational sensitivity of the models. These experiments provide a broader comparison of solution quality, geometric consistency, scale generalization, and robustness under changes in instance geometry.

C.1 TSP Behavioral Results

We compare L2C-Insert, LEHD, POMO, AM, and cluster-trained variants of POMO and LEHD across TSP sizes from 20 to 1000 and seven distribution shifts. We report the optimality gap relative to Concorde, the average number of edge crossings, and the convex-hull order violation rate. An optimal Euclidean TSP tour contains no edge crossings and visits convex-hull vertices in their cyclic order; therefore, these two metrics capture complementary forms of geometric inconsistency. Lower values are better, and the best result for each metric, including ties, is shown in bold.

Tables 3–8 reveal a clear separation between tour quality and geometric regularity. At smaller problem sizes, L2C-Insert and standard LEHD are generally competitive, whereas POMO and AM tend to exhibit larger optimality gaps. As the problem size increases, however, LEHD-Cluster becomes increasingly strong and consistently outperforms standard LEHD. This effect is especially pronounced on TSP-500 and TSP-1000, where LEHD-Cluster achieves the best or near-best optimality gaps across the tested distributions.

Interestingly, this improvement appears to stem from cluster-based training rather than from the architecture alone. LEHD-Cluster generalizes substantially better than standard LEHD as the problem size grows, including on unseen distributions. One plausible explanation is that cluster training encourages a more decompositional strategy, where the model solves spatially coherent regions as smaller subproblems before connecting them. We view this as a behavioral hypothesis rather than direct mechanistic evidence.

Cluster training also substantially improves the large-scale generalization of POMO, reducing its optimality gap across all seven distributions on TSP-500 and TSP-1000. Together with the LEHD-Cluster results, this suggests that clustered training can benefit both architectures by improving how they exploit spatial structure.

The geometric metrics provide a complementary view. POMO and POMO-Cluster can exhibit relatively low convex-hull violation rates despite substantially larger optimality gaps, indicating that preserving coarse geometric structure is not sufficient for producing a low-cost tour. AM shows a similar separation in several settings, where geometrically regular behavior does not necessarily translate into strong tour quality. Overall, the results suggest that large-scale generalization depends not only on preserving geometric constraints, but also on learning a solution strategy that can organize and coordinate decisions across increasingly large instances.

Table 3: Solution quality and geometric consistency under greedy decoding on TSP-20 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM			POMO (Cluster-Trained)			LEHD (Cluster-Trained)		
	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)
Clustered	0.55	0.06	3.83	0.49	0.07	4.86	9.41	0.49	8.22	26.68	2.99	51.24	3.37	0.18	3.00	**0.19**	**0.01**	**0.95**
Expansion	3.55	0.39	20.96	3.04	0.30	13.78	6.51	0.50	8.36	17.72	2.04	26.15	4.48	0.23	**2.00**	**0.93**	**0.07**	2.97
Explosion	2.67	0.26	9.99	1.13	0.08	3.66	5.62	0.34	6.11	9.70	0.72	13.48	3.83	0.37	5.00	**0.59**	**0.03**	**0.90**
Grid	2.22	0.21	8.90	0.96	0.07	3.16	5.33	0.31	5.75	8.12	0.52	10.34	3.63	0.16	2.00	**0.61**	**0.02**	**0.77**
Implosion	2.04	0.20	8.48	0.95	0.07	3.24	5.27	0.30	5.39	8.36	0.54	10.15	3.68	0.24	4.00	**0.62**	**0.02**	**0.77**
Mixed	2.42	0.25	11.08	1.44	0.13	5.44	8.34	0.66	9.39	12.91	1.22	18.56	6.25	0.43	4.00	**0.98**	**0.07**	**2.29**
Uniform	2.18	0.21	8.89	0.94	0.07	3.08	5.27	0.29	5.13	8.20	0.53	10.28	3.60	0.18	2.00	**0.61**	**0.03**	**1.01**

Table 4: Solution quality and geometric consistency under greedy decoding on TSP-50 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM			POMO (Cluster-Trained)			LEHD (Cluster-Trained)		
	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)
Clustered	0.65	0.05	2.24	0.91	0.14	7.10	5.26	0.27	2.31	33.42	6.57	41.24	1.72	0.04	**0.00**	**0.26**	**0.02**	0.42
Expansion	2.61	0.43	17.21	4.21	0.67	28.55	2.69	0.31	3.26	13.38	2.76	11.25	3.12	0.21	**2.00**	**0.94**	**0.10**	4.00
Explosion	**0.36**	**0.02**	0.75	0.55	0.05	1.41	1.37	0.09	**0.74**	5.71	0.24	1.93	2.19	0.11	1.00	0.66	0.05	0.95
Grid	**0.39**	**0.02**	0.51	0.50	0.03	0.74	0.85	**0.02**	0.07	4.36	0.12	0.57	2.34	0.05	**0.00**	0.73	0.04	0.97
Implosion	**0.38**	**0.02**	0.50	0.50	0.03	0.84	0.89	0.03	0.09	4.54	0.13	0.73	2.28	0.04	**0.00**	0.69	0.04	0.92
Mixed	**0.74**	**0.05**	1.34	0.89	0.13	2.66	3.31	0.41	2.85	6.81	0.46	3.13	4.25	0.19	**0.00**	1.15	0.13	2.31
Uniform	**0.37**	**0.01**	0.46	0.51	0.03	0.76	0.87	0.03	**0.07**	4.34	0.11	0.56	2.34	0.10	1.00	0.72	0.04	1.06

Table 5: Solution quality and geometric consistency under greedy decoding on TSP-100 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM			POMO (Cluster-Trained)			LEHD (Cluster-Trained)		
	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)
Clustered	1.06	0.13	3.17	1.49	0.36	12.71	7.00	1.19	8.26	41.40	13.61	39.81	2.61	0.23	**0.00**	**0.45**	**0.04**	0.70
Expansion	2.55	0.62	20.14	5.16	1.51	49.94	4.59	1.29	9.99	12.63	3.29	7.38	3.86	0.24	**1.00**	**1.09**	**0.16**	5.07
Explosion	**0.45**	**0.02**	**0.37**	0.63	0.08	1.37	2.12	0.46	3.49	6.55	0.45	0.47	2.75	0.19	1.00	0.75	0.09	1.28
Grid	**0.47**	**0.01**	0.19	0.56	0.03	0.42	0.90	0.07	0.07	4.32	0.19	0.03	3.20	0.24	**0.00**	0.85	0.06	1.19
Implosion	**0.46**	**0.02**	0.13	0.59	0.04	0.70	0.87	0.07	0.04	4.50	0.22	0.07	3.06	0.18	**0.00**	0.83	0.06	0.83
Mixed	**0.86**	**0.09**	0.95	1.03	0.24	3.96	4.77	1.66	7.96	7.24	0.59	**0.65**	4.66	0.82	1.00	1.41	0.26	3.87
Uniform	**0.46**	**0.02**	0.30	0.57	0.03	0.63	0.85	0.06	0.08	4.35	0.20	**0.03**	3.18	0.16	1.00	0.85	0.07	1.34

Table 6: Solution quality and geometric consistency under greedy decoding on TSP-200 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM			POMO (Cluster-Trained)			LEHD (Cluster-Trained)		
	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)
Clustered	2.39	0.48	6.74	2.61	0.71	18.70	14.59	7.83	22.10	33.81	15.57	21.70	7.72	1.99	**1.00**	**1.36**	**0.10**	2.06
Expansion	**0.86**	**0.10**	0.30	1.52	0.33	0.92	3.61	0.71	0.02	7.07	0.46	**0.00**	7.42	1.01	**0.00**	1.29	0.16	0.68
Explosion	**0.92**	**0.08**	0.62	1.08	0.09	0.92	4.03	0.67	0.14	7.16	0.43	**0.08**	7.97	0.90	3.00	1.41	0.14	2.02
Grid	1.01	0.09	2.32	**0.90**	**0.08**	1.86	5.23	1.39	0.52	8.64	0.53	**0.04**	8.92	1.64	1.00	1.10	0.16	2.26
Implosion	1.41	0.20	3.22	1.53	0.34	9.60	4.71	1.57	**0.74**	17.76	2.14	0.94	7.73	1.66	2.00	**0.89**	**0.10**	1.98
Mixed	1.24	**0.21**	2.96	**1.16**	0.27	5.16	7.70	4.94	7.38	9.46	1.23	0.26	8.39	2.54	**0.00**	1.29	0.30	4.34
Uniform	0.94	**0.09**	1.28	**0.88**	**0.09**	2.04	4.22	1.38	0.52	8.18	0.88	**0.04**	7.83	1.62	1.00	1.02	0.14	2.54

Table 7: Solution quality and geometric consistency under greedy decoding on TSP-500 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM			POMO (Cluster-Trained)			LEHD (Cluster-Trained)		
	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)
Clustered	5.58	6.45	41.06	5.41	3.86	47.98	43.54	70.07	59.50	51.78	50.90	25.04	24.71	15.83	**3.00**	**1.91**	**0.69**	14.36
Expansion	2.29	1.00	3.46	2.72	1.52	3.52	34.72	27.80	1.72	18.27	2.98	**0.00**	24.14	10.49	**0.00**	**1.75**	**0.61**	2.24
Explosion	2.16	0.77	7.26	2.05	0.76	5.46	33.14	22.83	3.16	18.78	3.12	**0.52**	24.92	9.31	2.00	**1.90**	**0.60**	5.60
Grid	2.29	1.35	17.86	1.77	0.87	19.22	26.94	30.47	8.88	21.14	2.79	**0.10**	22.19	9.16	3.00	**1.34**	**0.61**	12.78
Implosion	4.30	3.69	34.22	3.80	2.19	35.60	32.98	49.08	16.46	34.14	6.57	**1.36**	24.21	14.69	5.00	**1.27**	**0.51**	13.12
Mixed	2.89	2.61	19.62	2.30	1.61	19.46	35.67	62.64	34.04	21.33	4.72	**1.04**	25.15	16.29	8.00	**1.73**	**1.27**	17.30
Uniform	2.19	1.18	15.50	1.62	0.75	15.26	34.01	50.44	16.34	20.95	4.75	**0.96**	24.56	15.81	3.00	**1.36**	**0.60**	11.82

Table 8: Solution quality and geometric consistency under greedy decoding on TSP-1000 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM			POMO (Cluster-Trained)			LEHD (Cluster-Trained)		
	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)	Gap (%)	Cross.	Hull Viol. (%)
Clustered	9.45	29.76	69.00	9.58	13.44	65.00	66.42	212.07	75.00	75.53	188.73	40.00	37.36	38.42	**7.00**	**2.76**	**2.77**	48.00
Expansion	9.38	14.97	22.00	5.58	5.99	13.00	56.92	92.10	7.00	30.35	8.11	**0.00**	36.85	27.91	**0.00**	**2.37**	**1.94**	6.00
Explosion	4.65	4.78	41.00	3.83	3.60	11.00	52.55	73.73	4.00	31.19	8.31	**1.00**	38.35	29.75	3.00	**2.45**	**1.69**	15.00
Grid	4.50	7.54	52.00	3.56	4.11	39.00	39.61	86.19	22.00	33.42	8.62	**2.00**	32.19	26.03	11.00	**1.79**	**2.49**	35.00
Implosion	8.16	20.62	73.00	7.96	8.81	50.00	50.52	152.14	38.00	49.82	14.59	**1.00**	36.47	41.76	14.00	**1.85**	**1.98**	45.00
Mixed	5.25	13.33	45.00	4.49	7.16	49.00	55.75	188.51	71.00	35.33	11.76	**1.00**	38.32	44.73	11.00	**2.12**	**3.24**	34.00
Uniform	4.51	6.80	45.00	3.21	4.24	48.00	51.84	155.64	32.00	34.82	11.93	**7.00**	36.95	38.24	**7.00**	**1.76**	**2.02**	33.00

C.2 CVRP Behavioral Results

We compare L2C-Insert, LEHD, POMO and AM across CVRP sizes from 20 to 1000 and seven test distributions. We report the optimality gap relative to high-quality reference solutions obtained using a modern implementation of Hybrid Genetic Search (HGS), together with the geometric metrics defined in Section C.1. For CVRP, edge crossings and convex-hull order violations are evaluated separately within each vehicle route and then averaged across the routes and problem instances. Lower values are better for all metrics, and the best result in each setting, including ties, is shown in bold.

Tables 9–14 show that LEHD exhibits stronger scale generalization on the larger instances. It achieves the lowest optimality gap across all seven distributions on CVRP-200 and in six of the seven distributions on both CVRP-500 and CVRP-1000. In contrast, L2C-Insert generally produces fewer within-route crossings and lower convex-hull violation rates, indicating greater local geometric consistency.

Table 9: Solution quality and within-route geometric consistency under greedy decoding on CVRP-20 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM		
	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)
Clustered	6.99	**0.02**	**1.42**	**5.02**	0.04	3.47	71.63	0.27	16.87	14.27	0.24	18.84
Expansion	6.65	**0.02**	**1.68**	**5.59**	0.04	3.45	66.78	0.21	14.07	12.97	0.21	16.50
Explosion	6.80	**0.01**	**1.10**	**5.76**	0.04	3.11	66.77	0.18	12.36	12.72	0.17	14.15
Grid	6.91	**0.01**	**1.15**	**5.76**	0.03	2.83	65.97	0.15	10.92	12.71	0.17	13.97
Implosion	6.98	**0.01**	**1.16**	**5.82**	0.03	2.96	66.68	0.16	11.14	12.80	0.17	13.97
Mixed	7.17	**0.02**	**1.31**	**5.35**	0.04	2.98	60.44	0.18	11.94	13.54	0.20	15.22
Uniform	6.87	**0.01**	**1.18**	**5.71**	0.03	2.93	66.29	0.16	11.07	12.68	0.17	13.80

Table 10: Solution quality and within-route geometric consistency under greedy decoding on CVRP-50 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM		
	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)
Clustered	5.82	**0.02**	**1.27**	**5.19**	0.04	2.90	13.76	0.18	12.49	11.96	0.31	19.25
Expansion	5.24	**0.02**	**1.27**	**4.73**	0.04	2.99	11.89	0.16	11.13	9.83	0.25	16.55
Explosion	5.30	**0.01**	**0.80**	**4.80**	0.03	2.37	10.40	0.13	9.18	9.12	0.19	13.36
Grid	5.29	**0.01**	**0.68**	**5.01**	0.03	2.19	10.17	0.12	8.45	8.72	0.17	11.53
Implosion	5.33	**0.01**	**0.73**	**5.00**	0.03	2.33	10.26	0.12	8.37	8.74	0.17	12.06
Mixed	5.50	**0.01**	**0.93**	**4.51**	0.03	2.53	16.40	0.13	9.31	9.95	0.23	14.27
Uniform	5.36	**0.01**	**0.68**	**5.01**	0.03	2.18	10.22	0.11	8.26	8.74	0.17	11.85

Table 11: Solution quality and within-route geometric consistency under greedy decoding on CVRP-100 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM		
	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)
Clustered	**4.96**	**0.02**	**1.38**	5.05	0.05	3.48	5.49	0.12	8.30	12.91	0.45	23.67
Expansion	4.38	**0.02**	**1.17**	4.43	0.04	3.26	**3.91**	0.10	6.76	9.70	0.32	17.83
Explosion	4.12	**0.01**	**0.77**	4.28	0.03	2.46	**3.37**	0.07	4.61	8.63	0.23	13.05
Grid	3.87	**0.01**	**0.54**	4.16	0.03	1.92	**2.98**	0.06	3.59	7.65	0.17	9.98
Implosion	3.91	**0.01**	**0.60**	4.29	0.03	2.02	**3.03**	0.06	3.70	7.76	0.17	10.38
Mixed	4.20	**0.01**	**0.96**	4.09	0.04	2.62	**3.72**	0.08	4.78	8.95	0.29	14.38
Uniform	3.88	**0.01**	**0.56**	4.18	0.03	1.88	**2.98**	0.06	3.73	7.67	0.17	9.87

Table 12: Solution quality and within-route geometric consistency under greedy decoding on CVRP-200 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM		
	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)
Clustered	5.81	**0.05**	**2.52**	**5.11**	0.09	5.27	10.50	0.31	13.50	21.96	0.93	37.94
Expansion	5.97	**0.05**	**2.44**	**4.34**	0.07	4.81	9.62	0.33	12.47	13.77	0.58	24.98
Explosion	5.29	**0.03**	**1.59**	**3.84**	0.05	3.53	9.26	0.28	10.58	12.09	0.39	17.16
Grid	4.89	**0.03**	**1.42**	**3.43**	0.04	2.65	9.00	0.23	8.58	10.52	0.26	12.05
Implosion	5.15	**0.07**	**3.17**	**3.78**	**0.07**	4.18	9.17	0.32	11.66	13.41	0.41	15.71
Mixed	5.07	**0.03**	**1.79**	**3.57**	0.07	3.89	9.51	0.34	13.69	11.91	0.54	20.32
Uniform	4.90	**0.03**	**1.35**	**3.47**	0.04	2.73	9.08	0.24	8.91	10.66	0.27	12.11

Table 13: Solution quality and within-route geometric consistency under greedy decoding on CVRP-500 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM		
	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)
Clustered	**6.00**	**0.18**	**6.27**	6.72	0.34	14.97	59.38	3.67	49.81	74.56	2.07	48.33
Expansion	5.05	**0.15**	**5.72**	**4.80**	0.25	13.20	52.37	4.25	54.58	33.24	1.86	48.68
Explosion	4.75	**0.11**	**4.16**	**4.08**	0.19	10.38	49.54	3.32	51.04	28.88	1.44	40.05
Grid	4.24	**0.08**	**3.30**	**3.36**	0.14	8.63	39.47	2.52	47.73	21.74	0.96	30.62
Implosion	4.26	**0.18**	**5.84**	**3.58**	0.27	10.93	44.40	3.22	51.94	33.76	1.08	28.25
Mixed	5.14	**0.15**	**5.79**	**4.25**	0.28	12.01	61.65	3.38	51.09	26.51	1.58	38.69
Uniform	4.18	**0.08**	**3.32**	**3.31**	0.14	8.25	40.03	2.59	47.90	21.85	1.00	30.66

Table 14: Solution quality and within-route geometric consistency under greedy decoding on CVRP-1000 across seven test distributions.

Distribution	L2C-Insert			LEHD			POMO			AM		
	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)	Gap (%)	Cross./Route	Hull Viol. (%)
Clustered	**10.77**	1.04	**21.95**	10.84	**1.02**	26.28	265.92	19.10	75.78	303.36	1.82	22.42
Expansion	6.58	**0.72**	**16.67**	**6.08**	0.82	28.04	308.93	19.57	73.11	118.61	2.64	37.37
Explosion	6.83	**0.49**	**13.38**	**5.24**	0.54	24.21	304.22	17.29	76.34	123.33	1.92	32.32
Grid	5.78	**0.30**	**9.32**	**4.25**	0.42	20.62	303.32	18.03	81.67	61.48	1.94	39.11
Implosion	5.36	**0.60**	**14.41**	**4.21**	0.77	23.57	289.05	18.20	80.75	109.97	1.52	26.42
Mixed	8.58	1.14	22.14	**6.25**	**0.86**	**21.93**	236.46	12.30	65.42	61.92	2.92	40.06
Uniform	5.42	**0.31**	**8.81**	**4.02**	0.40	20.81	301.74	17.49	80.67	63.87	2.01	38.81

C.3 SENSITIVITY

We evaluate the rotational robustness of L2C-Insert, LEHD, POMO, and AM. For model m and instance i, let $g_{i,\alpha}^{(m)}$ denote the optimality gap after rotating the instance by angle α. We consider $\mathcal{A} = \{0°, 60°, 120°, 180°, 210°, 270°\}$. The rotation sensitivity of each instance is defined as $s_i^{(m)} = \max_{\alpha\in\mathcal{A}} g_{i,\alpha}^{(m)} - \min_{\alpha\in\mathcal{A}} g_{i,\alpha}^{(m)}$. We report the mean sensitivity over the M evaluated instances as $S^{(m)} = \frac{1}{M}\sum_{i=1}^{M} s_i^{(m)}$.

Lower values indicate greater rotational robustness. Tables 15 and 16 show that L2C-Insert generally has the lowest sensitivity at smaller problem sizes, while LEHD remains robust at larger scales. On TSP-500 and TSP-1000, LEHD achieves the lowest sensitivity in five and four of the seven distributions, respectively. POMO is generally more sensitive than LEHD, while AM exhibits particularly high sensitivity in several settings. The best result in each row, including ties, is shown in bold.

Table 15: Rotation sensitivity across test distributions for TSP-20, TSP-50, and TSP-100.

Distribution	TSP-20				TSP-50				TSP-100			
	L2C-Insert	LEHD	POMO	AM	L2C-Insert	LEHD	POMO	AM	L2C-Insert	LEHD	POMO	AM
Clustered	**0.91**	1.17	22.97	61.24	**1.08**	1.77	9.03	72.37	**1.56**	2.39	11.61	79.98
Expansion	13.21	**7.69**	19.82	42.53	**5.00**	6.52	8.64	22.26	**4.25**	5.62	12.76	20.04
Explosion	7.70	**3.75**	17.03	32.03	**0.77**	1.23	4.10	11.90	**0.83**	1.16	6.30	8.75
Grid	6.58	**2.81**	16.40	26.12	**0.82**	1.11	2.44	6.90	**0.81**	1.03	1.76	4.81
Implosion	6.21	**2.84**	16.51	27.93	**0.84**	1.12	2.55	7.15	**0.78**	1.09	1.80	4.89
Mixed	6.79	**4.53**	21.64	34.33	**1.68**	2.00	11.78	11.65	**1.52**	1.82	18.09	6.44
Uniform	6.79	**2.97**	16.21	27.41	**0.86**	1.12	2.46	7.16	**0.80**	1.03	1.71	4.84

Table 16: Rotation sensitivity across test distributions for TSP-200, TSP-500, and TSP-1000.

Distribution	TSP-200				TSP-500				TSP-1000			
	L2C-Insert	LEHD	POMO	AM	L2C-Insert	LEHD	POMO	AM	L2C-Insert	LEHD	POMO	AM
Clustered	**1.94**	2.38	16.87	96.41	3.03	**2.93**	20.59	101.45	**3.45**	3.46	24.60	113.51
Expansion	**4.82**	7.18	5.66	7.46	24.23	11.62	15.75	**9.25**	92.33	12.44	14.95	**9.93**
Explosion	**1.46**	2.55	6.14	6.22	**4.44**	6.34	16.15	6.69	21.71	9.71	14.35	**6.42**
Grid	**1.05**	1.10	6.00	4.22	1.89	**1.56**	10.30	4.21	2.65	**1.94**	9.28	3.72
Implosion	**1.67**	1.87	6.68	174.96	2.67	**2.36**	12.24	193.60	3.30	**2.64**	10.32	269.56
Mixed	**1.53**	**1.53**	19.15	5.06	2.55	**2.05**	23.34	5.28	2.99	**2.30**	28.46	4.82
Uniform	**1.18**	1.20	5.65	5.04	1.92	**1.76**	12.57	5.30	2.89	**2.46**	10.36	4.99

D Extended Mechanistic Interpretability Results

D.1 Controlling for Nearest-Neighbor Heuristics in LEHD

To examine whether LEHD's multi-step future-action probe performance is primarily driven by a simple nearest-neighbor heuristic, we restrict the evaluation to decoding states where LEHD's own greedy next action is not the nearest currently unvisited node. Results for the final two decoder layers, L5 and L6, are reported in Table 17.

Probe accuracy decreases under this restriction, but remains substantial for multiple future steps. On TSP-500, for instance, L5 achieves 56.2% Top-1 accuracy at $h = 2$ and 30.3% at $h = 3$, even though the model's immediate action is explicitly non-nearest-neighbor. The same qualitative pattern holds for TSP-50 and TSP-100. Thus, the future-action signal accessible from LEHD's decoder representations is not reducible to a simple nearest-neighbor rule. These results indicate that the future-action information encoded in LEHD's decoder representations cannot be explained solely by a simple nearest-neighbor heuristic and instead reflects richer information about the model's subsequent trajectory.

Table 17: Top-1 future-node probe accuracy (%) for LEHD on TSP-50, TSP-100, and TSP-500. *Own-Trajectory Future* evaluates prediction of LEHD's own future nodes from its greedy decoding trajectory. *Own-Trajectory Future (Non-NN)* restricts evaluation to states where LEHD's greedy next action is not the nearest currently unvisited node.

		Own-Trajectory Future						Own-Trajectory Future (Non-NN)					
Size	Layer	h=1	h=2	h=3	h=4	h=5	h=6	h=1	h=2	h=3	h=4	h=5	h=6
TSP-50	L5	95.6	86.5	60.5	32.4	22.5	17.2	85.4	70.5	40.2	30.6	19.6	16.0
TSP-50	L6	99.8	84.2	54.1	31.0	21.4	15.8	99.4	67.6	38.6	27.9	18.9	14.8
TSP-100	L5	95.6	86.0	58.1	32.9	22.4	16.7	83.4	68.4	39.4	30.1	19.7	15.2
TSP-100	L6	99.9	82.4	50.4	31.2	20.9	15.5	99.8	65.1	35.5	27.7	19.0	14.3
TSP-500	L5	94.5	77.7	43.6	28.2	20.4	15.4	78.2	56.2	30.3	22.5	17.4	13.7
TSP-500	L6	99.9	76.0	40.0	25.0	17.9	13.8	99.8	56.6	28.3	19.8	14.8	11.1

D.2 Behavioral-Causal Node Perturbation

In addition to the counterfactual steering experiment described in Section 3.3, we conduct a second causal experiment that does not rely on probes or manual modifications to the model's embeddings. Our setup compares the model's greedy rollout branch, $a_1, a_2, \ldots$, with the greedy rollout starting from the second-highest-probability node, $a'_1, a'_2, \ldots$. We perturb the coordinates of nodes $a'_2, a'_3, \ldots, a'_5$, and then observe how the logit gap between a_1 and a'_1 changes at each step t:

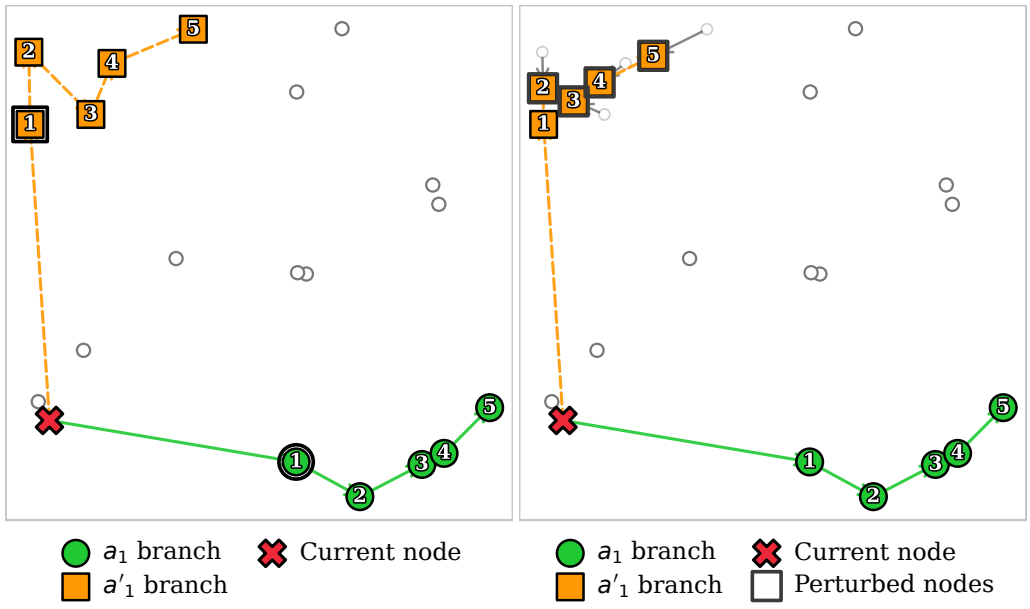
$$\Delta G_t = [z_{\text{steered}}(a'_1) - z_{\text{steered}}(a_1)] - [z_{\text{clean}}(a'_1) - z_{\text{clean}}(a_1)]\,,$$

where $z_{\text{steered}}(\cdot)$ and $z_{\text{clean}}(\cdot)$ denote the logits under the perturbed and clean inputs, respectively.

We consider four perturbation categories:

- The **toward** category shortens the distance between consecutive nodes a'_i and a'_{i-1} using an interpolation coefficient α:
$$r_{a'_i} \leftarrow (1-\alpha) r_{a'_i} + \alpha r_{a'_{i-1}},$$
where r_a denotes the location vector of node a.
- Similarly, the **away** category increases the distance between consecutive nodes a'_i and a'_{i-1} using an extrapolation coefficient α.
- The **random direction** category perturbs the node in a random direction.
- The **random node** category perturbs a node in a far horizon, e.g., a'_{10}.

In most scenarios, there is only one plausible next node, with selection probability ≈ 1, and the second-best node is clearly poor. We therefore restrict our analysis to scenarios in which the second-most-probable node has a reasonable chance, defined as selection probability > 0.20. This makes the perturbations more meaningful. A sample visualization is presented in Figure 6



(a) Before perturbation (b) After perturbation

Figure 6: A sample visualization of node perturbation. **(a)** Shows the original instance. **(b)** Shows the perturbed instance.

In Figure 7, we observe that the gap between the most probable node, i.e., the original next node, and the second-highest-probability node in LEHD decreases in the **toward** category, where the distance between consecutive nodes along that horizon is shortened. This indicates that LEHD's preference shifts toward the second-most-probable branch. In contrast, we observe no meaningful change for the other perturbation categories, signaling that LEHD's decision remains relatively unchanged in those cases. Interestingly, this perturbation becomes even more effective as the model progresses through decoding.

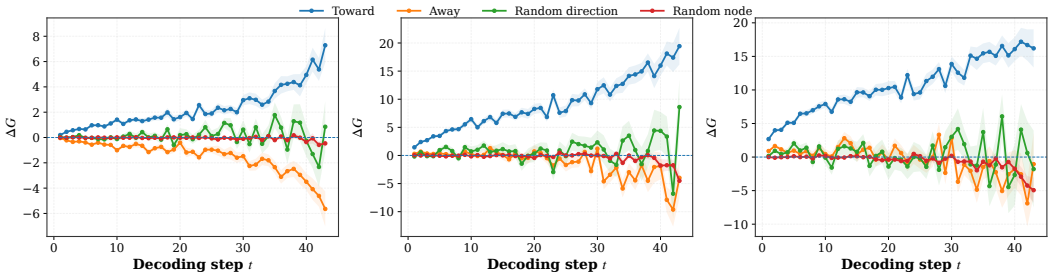


Figure 7: LEHD ΔG with probability threshold.

We also conducted the same experiment for POMO, with results shown in Figure 8. Although the **toward** category also produces a larger effect than the other categories, its value is much smaller than that of LEHD. This shows that LEHD is much more sensitive to the locations of nodes beyond the immediate next node than POMO is, highlighting the difference in look-ahead capabilities discussed in Section 5. Moreover, the effect of perturbation is relatively constant at every decoding step in POMO, unlike in LEHD, further highlighting POMO's lack of decoding-step sensitivity.

These experiments, together with the counterfactual steering experiment presented in Section 3.3, provide causal evidence for the look-ahead mechanism in LEHD.

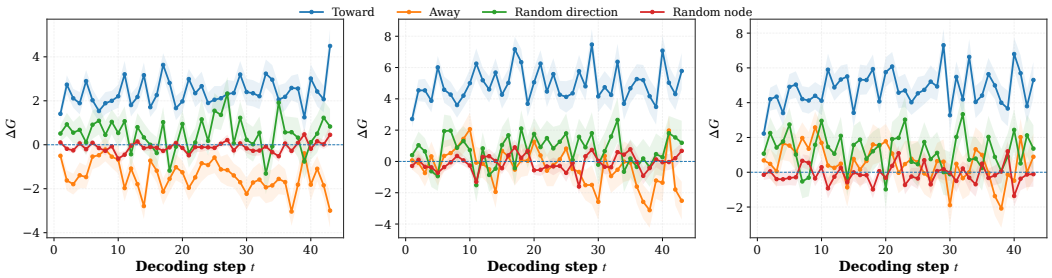


Figure 8: POMO ΔG with probability threshold.

D.3 Assessing Off-Manifold Effects in Probe-Guided Steering

A potential concern with the probe-guided counterfactual steering experiment in Section 3.3 is that the observed behavioral changes may result from pushing internal representations away from the model's naturally occurring representation manifold, rather than from selectively modifying the future-action information under investigation. To assess the extent of such off-manifold perturbations, we measure three complementary representation-space diagnostics: the ratio of steered to clean representation norms, the cosine similarity between steered and clean representations, and the relative norm of the perturbation. These quantities are computed for the targeted future-node representations across the intervened decoder layers and horizons. Table 18 reports these diagnostics across steering strengths to assess whether the interventions substantially alter the model's natural representation geometry.

Table 18: Representation-space diagnostics for probe-guided counterfactual steering on Uniform TSP-100 across future horizons and decoder layers.

α	Norm ratio	Cosine similarity	Relative perturbation
3	1.031 ± 0.002	0.9873 ± 0.0004	0.151 ± 0.002
5	$\mathbf{1.060 \pm 0.003}$	$\mathbf{0.9676 \pm 0.0008}$	$\mathbf{0.250 \pm 0.003}$
10	1.156 ± 0.006	0.8959 ± 0.0022	0.495 ± 0.007

D.4 Start-Node Navigation in AM

Although all models are evaluated in a single-trajectory setting, AM differs from POMO and LEHD in how the starting node is selected. While POMO and LEHD begin from an arbitrary starting node, AM selects its starting node through its learned policy. At the initial decoding step, AM uses two learnable placeholder embeddings for the first and current nodes, enabling start selection to be learned jointly with subsequent route construction. AM and POMO also differ in their decoder conditioning: AM concatenates the first- and current-node embeddings and incorporates a separate graph-level representation, whereas POMO combines their projected embeddings through addition. These differences in start selection and learned decoder conditioning may contribute to AM's greater sensitivity to start-node interventions.

As shown in Figure 9, start-node patching causes substantially smaller probability drops when node 0 is prescribed as the starting node rather than selected by AM's policy. This suggests that learned start selection may make the start-node representation more influential in subsequent routing decisions, although the two settings also produce different tour trajectories. One possible explanation for the contrasting trends in AM and LEHD is that AM relies more on start-node information during early and middle tour construction, whereas LEHD may use it more heavily as the tour approaches completion.

Under *maximum-angle donor selection without distance constraints*, policy-selected starts yield lower LCS and higher Rev. LCS than forced starts in Phase I (Table 19), but this pattern is less consistent in Phase II (Table 20). Unlike LEHD (Table 1), AM retains higher forward- than reverse-

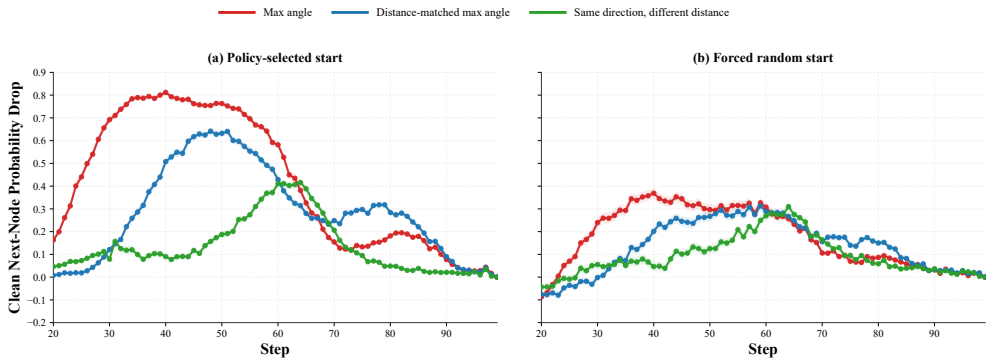


Figure 9: **Start-node intervention sensitivity in AM on Uniform TSP-100.** (a) Policy-selected start; (b) forced start at node 0, followed by greedy decoding. Previously visited donors are selected by maximum angular separation (red), maximum angle with matched distance (blue), or similar direction with different distance (green). The y-axis shows the mean drop in clean next-action probability after patching.

order similarity even at large angles. This may suggest a policy-dependent route-anchoring role in AM, compared with a possible global-navigation reference in LEHD.

Table 19: AM rollout-order similarity on Uniform TSP-100 under maximum-angle donor selection. Phase I: steps 20–45.

Angle Range	Policy Start		Forced Start	
	LCS	Rev. LCS	LCS	Rev. LCS
0°–30°	**0.626**	0.159	**0.680**	0.137
30°–60°	**0.564**	0.183	**0.583**	0.172
60°–90°	**0.464**	0.215	**0.522**	0.191
90°–120°	**0.369**	0.250	**0.484**	0.204
120°–150°	**0.351**	0.263	**0.435**	0.221
150°–180°	**0.380**	0.261	**0.437**	0.227

Table 20: AM rollout-order similarity on Uniform TSP-100 under maximum-angle donor selection. Phase II: steps 46–99.

Angle Range	Policy Start		Forced Start	
	LCS	Rev. LCS	LCS	Rev. LCS
0°–30°	–	–	–	–
30°–60°	**0.782**	0.138	**0.748**	0.145
60°–90°	**0.726**	0.159	**0.711**	0.163
90°–120°	**0.680**	0.180	**0.689**	0.175
120°–150°	**0.708**	0.176	**0.688**	0.178
150°–180°	**0.760**	0.172	**0.685**	0.182

E VISUALIZATION

We present additional visualizations of tour-construction patterns, routing behaviors, and internal representations of neural solvers.

E.1 LATENT-SPACE ORGANIZATION

Figure 10 provides a qualitative view of how LEHD organizes node-role representations across decoder layers. A single PCA basis is jointly fitted across all decoder layers, allowing direct comparison of their representations in a shared latent space. The reported explained-variance ratios correspond to this joint PCA. The faint markers show instance-level representations, while the larger markers denote their corresponding means. From Layer 2 onward, current-node representations from different instances become concentrated in a compact shared region, consistent with the cross-instance node alignment reported in the main text. In contrast, the start-node representations remain more dispersed in the earlier layers, but their distinct region becomes increasingly apparent from Layer 4 onward, suggesting that the decoder progressively separates the start node as a dedicated global reference. The mean future-node representations also exhibit an approximately ordered trajectory, particularly in the intermediate layers.

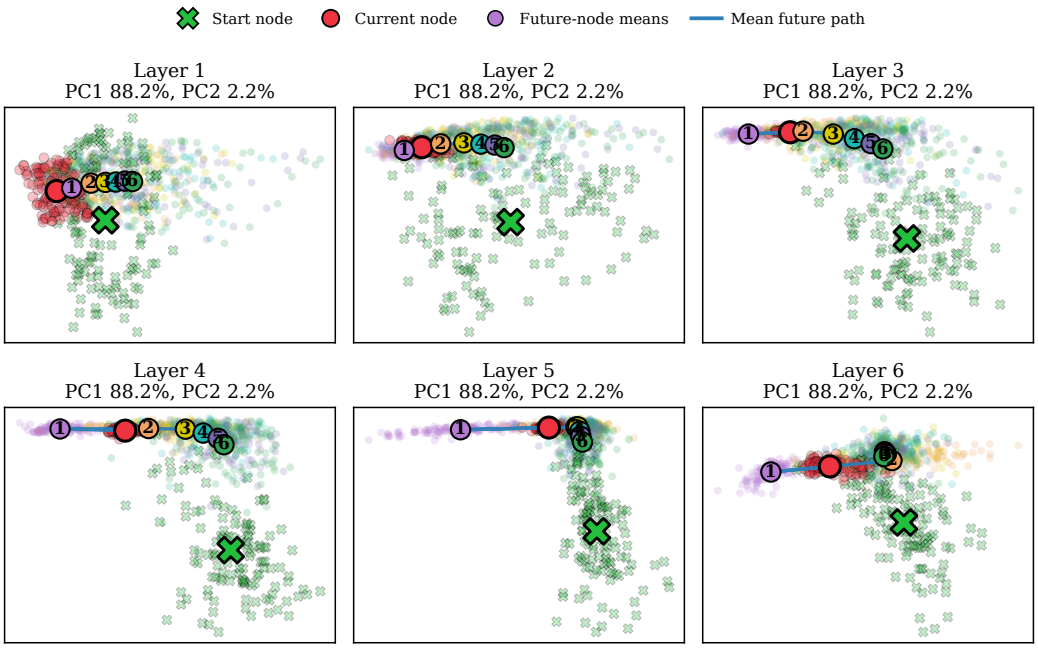


Figure 10: **Layer-wise PCA of LEHD node-role representations.** Faint points show instance-level representations across 128 Uniform TSP-50 instances, while larger markers indicate their means. Numbered markers denote LEHD's next six actions under greedy decoding.

E.2 GEOMETRIC TOUR-CONSTRUCTION PATTERNS

Figures 11 and 12 provide representative examples of POMO and AM rollouts across increasing problem sizes. Node colors indicate normalized onion depth, with outer nodes assigned lower values and interior nodes higher values. In both models, the highlighted segments reveal outer-to-deep-to-outer excursions during tour construction. In POMO, this pattern appears clearly already on TSP-50 and persists, becoming more pronounced as the problem size increases to TSP-100 and TSP-200, consistent with the oscillatory behavior quantified by the onion-depth analysis. The corresponding AM examples show that similar excursions can also arise under the same visualization, providing a qualitative comparison of geometric tour-construction patterns across the two models, without implying that the frequency or strength of these excursions is the same in AM and POMO.

Figure 13 qualitatively compares the radial organization of POMO, LEHD, and AM routes on Uniform CVRP-200 and CVRP-500. Among the selected high-backtracking examples, POMO exhibits the most pronounced radial reversals, producing intertwined trajectories that repeatedly move toward and away from the depot. AM shows an intermediate pattern, with several routes displaying radial reversals alongside more structured outward and return segments. In contrast, LEHD exhibits the most regular radial organization in these examples, with routes generally extending toward their peak-radius nodes before returning to the depot. The differences become particularly visible on CVRP-500, where the longer routes make the contrasting geometric structures easier to observe. These visualizations illustrate the differences in radial backtracking examined in the quantitative analysis.

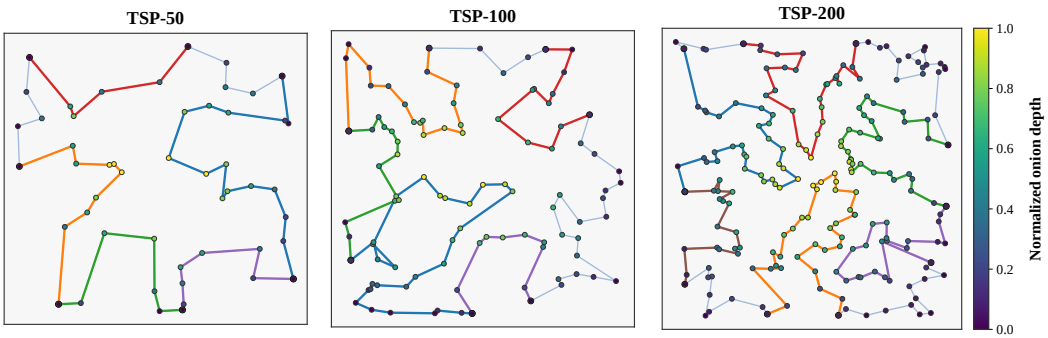


Figure 11: **Onion-depth structure of POMO tours.** Representative Uniform TSP-50, TSP-100, and TSP-200 rollouts illustrate POMO's repeated oscillation between outer and deeper onion layers. Colored segments highlight example outer-to-deep-to-outer excursions, while node colors indicate normalized onion depth.

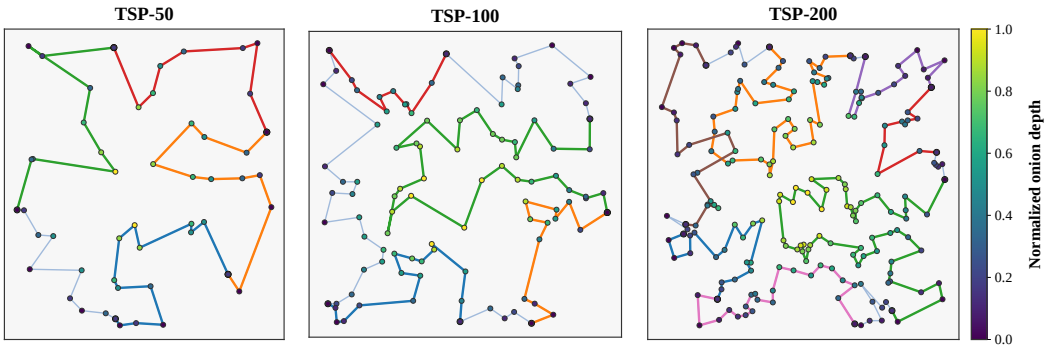
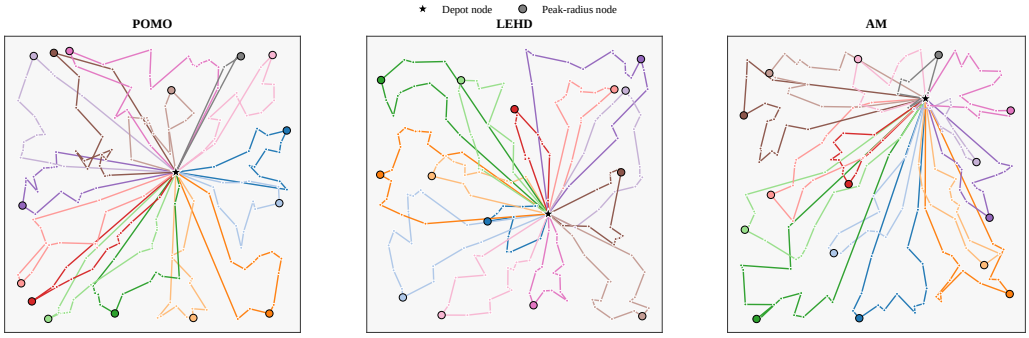


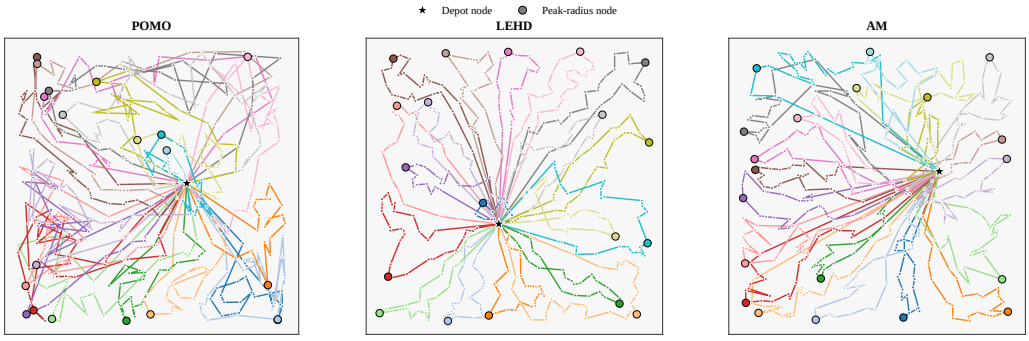
Figure 12: **Onion-depth structure of AM tours.** Selected Uniform TSP-50, TSP-100, and TSP-200 rollouts illustrate outer-to-deep-to-outer excursions in AM. Colored segments highlight example excursions, while node colors indicate normalized onion depth.

E.3 Attention Patterns During Decoding

In Figure 4a of the main text, we visualized the attention heatmap for a single layer of LEHD to demonstrate that the model attends more strongly to nodes scheduled for future visits than to other unvisited nodes. In Figure 14, we provide a detailed layer-wise breakdown of this behavior across all decoder layers. The upcoming nodes in the ground-truth sequence are highlighted with red dashed lines. For visual clarity, self-attention from the current node to itself has been omitted. Notably, the attention distribution is more diffuse in the initial layer and becomes increasingly focused on upcoming nodes in subsequent layers, suggesting that future-action planning primarily occurs in the deeper decoding layers. Additionally, an animated visualization of the step-wise attention dynamics across layers is provided in the supplementary media.



(a) Uniform CVRP-200.



(b) Uniform CVRP-500.

Figure 13: **Radial route structure of POMO, LEHD, and AM on CVRP.** Selected Uniform CVRP-200 and CVRP-500 solutions are shown, with POMO on the left, LEHD in the center, and AM on the right. Each vehicle route is displayed in a distinct color; black stars indicate depot locations, and enlarged circular markers identify the farthest customer from the depot along each route. In these selected examples, radial backtracking is most visually pronounced in POMO, followed by AM, while LEHD exhibits more regular outward-then-inward route structures.

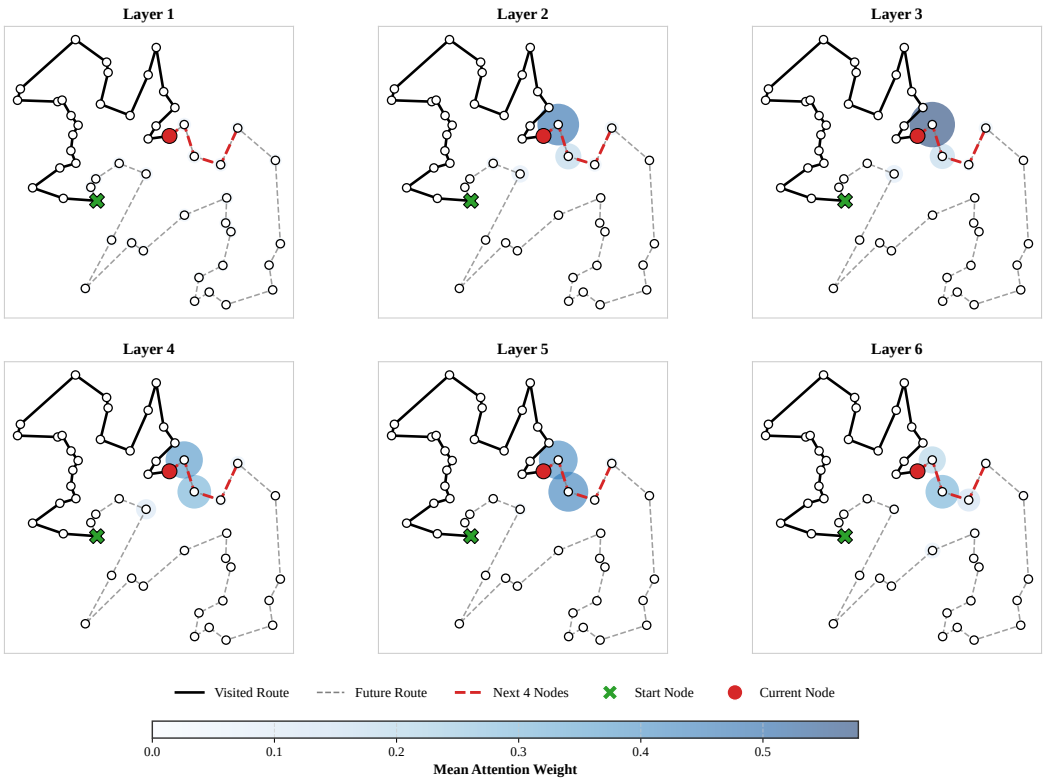


Figure 14: **Layer-wise attention heatmaps of the current node during decoding.** Each panel illustrates the attention weight distribution assigned by the current node (query) to remaining unvisited nodes (keys) for each decoder layer at a representative decoding step. Red dashed lines denote the nodes actually visited in subsequent steps.

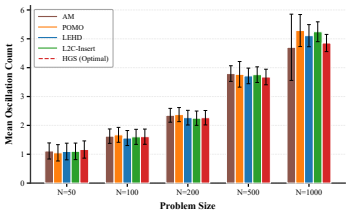
F CVRP RESULTS

In this section, we extend the interpretability experiments presented in Section 3 to the Capacitated Vehicle Routing Problem (CVRP). Unlike the TSP, where a solution consists of a single continuous tour, CVRP solutions comprise multiple distinct routes that each originate and terminate at a central depot. Here, we evaluate whether the mechanistic and geometric properties observed in TSP generalize to individual routes within CVRP solutions.

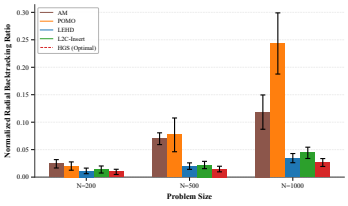
F.1 GEOMETRIC TRAJECTORY ANALYSIS

For the TSP, we established that POMO constructs tours via an oscillating clockwise progression. However, because individual routes in CVRP are substantially shorter than full TSP tours, even on large problem sizes, the average number of deep onion-layer oscillations per route is naturally low across all methods, rendering this metric less effective for distinguishing geometric biases (Figure 15a).

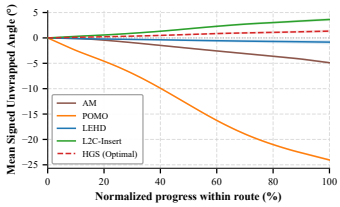
To capture radial geometric distortion within shorter individual routes, we first identify the node furthest from the depot within a given route and use it as an anchor to bisect the route into two segments. For the first segment, we compute the depot distance of each node and count how often the distance *decreases* between consecutive steps. For the second segment, we count how often the distance *increases*. Intuitively, this measures the degree of radial back-and-forth distortion along the route. As shown in Figure 15b and visually illustrated in Figure 13, POMO exhibits higher radial distortion than LEHD, L2C-Insert, and HGS, while AM shows comparable distortion at smaller scales but substantially less at N=1000. Furthermore, POMO maintains a consistent clockwise progression during route construction, mirroring its TSP behavior (Figure 15c). AM also exhibits a milder clockwise tendency on CVRP, in contrast to its counterclockwise progression on TSP.



(a) Mean onion-layer oscillations



(b) Radial route distortion



(c) Angular orientation progression

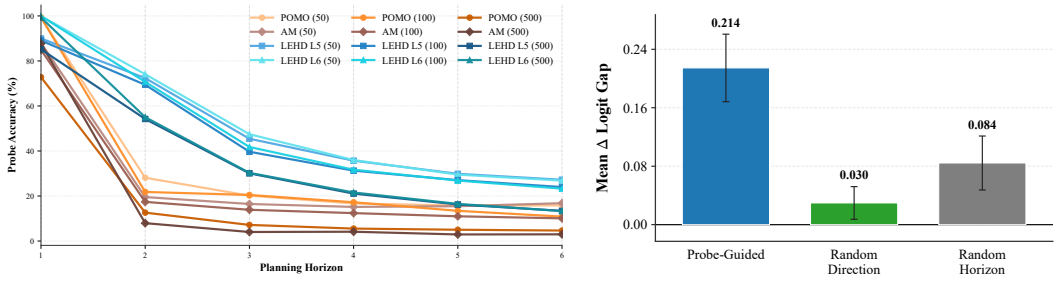
Figure 15: **Geometric trajectory properties on CVRP routes. (a)** Average onion-layer oscillations across problem scales; values remain low due to short individual route lengths. **(b)** Step-wise radial distortion along individual routes, showing heightened back-and-forth movement in POMO compared to baselines, with AM exhibiting intermediate distortion at larger scales. **(c)** Mean current-to-depot angle relative to each route's first edge, averaged within instances and then across instances, showing POMO's persistent clockwise progression and a weaker clockwise tendency in AM.

F.2 FUTURE-ACTION PLANNING PROBES

In Section 5, we showed that LEHD's internal representations encode multi-step future node sequences for the TSP. To determine whether this look-ahead capability extends to CVRP, we trained linear probes to predict upcoming nodes within the active route. For CVRP, the probe ranks all unvisited customers, regardless of their immediate feasibility under the remaining vehicle capacity.

We exclude horizons that cross a depot return, ensuring that each evaluated target belongs to the same route as the current customer. As shown in Figure 16a, LEHD demonstrates strong predictive accuracy over short future horizons, closely replicating the planning characteristics observed on TSP. In contrast, both POMO and AM retain only limited horizon-predictive information, with probe accuracy dropping sharply beyond one-step prediction.

To investigate whether these future-action representations also influence the current decision, we extend our probe-guided steering experiment to CVRP. At each selected state, we identify the model's preferred next customer H_1 and its second-best alternative H_1', then counterfactually force H_1' and collect subsequent customers $H_2', \ldots, H_5'$ within the same vehicle route. We intervene on the representations of these future customers without directly modifying H_1 or H_1'. As shown in Figure 16b, steering LEHD's future-node representations increases the logit gap in favor of H_1', with a larger mean effect than either the random-direction or disjoint-horizon control. These findings provide evidence that LEHD's accessible future-action information is causally connected to its current routing decisions in CVRP.



(a) Future-action linear probing accuracy

(b) Probe-guided steering

Figure 16: **Future-action representations and causal steering on CVRP. (a)** Linear probing accuracy across planning horizons for LEHD, POMO, and AM, revealing differences in the accessibility of future-action information from decoder representations. **(b)** Change in the logit gap under probe-guided steering and control interventions on CVRP-100, measuring the effect of modifying future-node representations on the current decision.

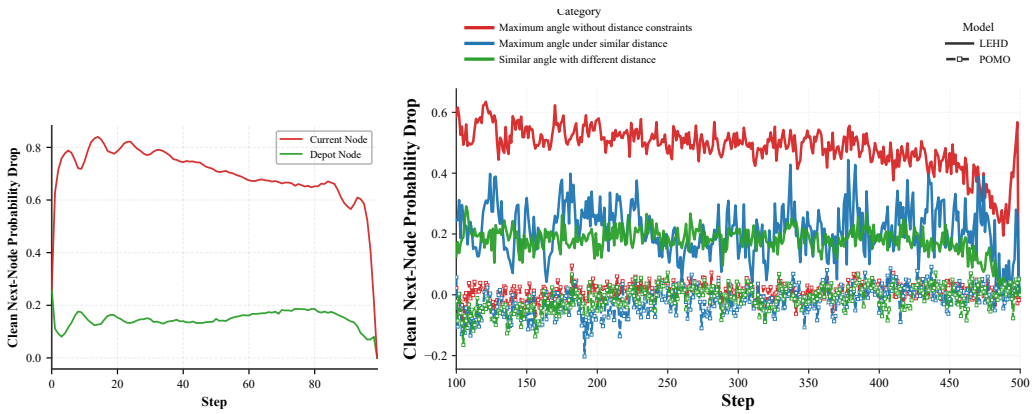
F.3 Current and Start Node Contribution

We next extend our causal intervention studies to CVRP routes. As in the TSP setting, similar causal patterns emerge: Figure 17a reveals that LEHD's immediate next-node predictions are far more sensitive to mean-ablating the current node representation than the depot node, while Figure 17b confirms that the angular orientation of the depot node guides global route navigation.

Interestingly, while the Longest Common Subsequence (LCS) trend under depot-node interventions decreases as the donor angle range widens (matching the TSP pattern in Table 1), the absolute LCS values are substantially lower for CVRP (Table 21). Intervening on the depot node often forces models to switch to entirely different customer clusters, resulting in non-overlapping route memberships and low sequence alignment. Furthermore, because CVRP routes are strictly constrained by vehicle capacity, LEHD cannot rely solely on start-node angular orientation to minimize return costs; it must simultaneously optimize capacity constraints, keeping Reverse LCS values consistently low across all donor angle ranges.

F.4 Node-Role Representation Alignment

A central finding for the TSP was that LEHD projects current and start node representations into canonical, shared regions of the embedding space across decoding steps and instances (Table 2). We evaluate this cross-instance representation alignment for CVRP in Table 22. The results confirm that the same canonical representation mechanism operates in CVRP: intermediate and late decoder layers maintain high cosine similarities for the current and depot nodes compared to randomly selected customer nodes.



(a) LEHD mean ablation impact

(b) POMO vs. LEHD start-node patching

Figure 17: **Causal interventions on node representations in CVRP. (a)** Probability drop in clean next-node predictions following mean ablation of current versus depot node representations in LEHD. **(b)** Effect of start/depot-node activation patching under three geometric donor conditions, comparing directional sensitivity in LEHD and POMO.

Table 21: **CVRP-500 route rollout-order similarity under depot-node activation patching.** Results evaluate maximum-angle donor selection without distance constraints. LCS measures forward-order sequence agreement with clean rollouts, while Rev. LCS measures agreement with reversed clean rollouts.

Angle Range	LEHD		POMO	
	LCS	Rev. LCS	LCS	Rev. LCS
0°–30°	**0.496**	0.123	**0.168**	0.079
30°–60°	**0.318**	0.126	**0.108**	0.065
60°–90°	**0.221**	0.117	**0.080**	0.064
90°–120°	**0.155**	0.107	**0.080**	0.071
120°–150°	**0.117**	0.107	**0.104**	0.083
150°–180°	**0.115**	0.111	**0.151**	0.118

Table 22: **Layer-wise cross-instance alignment of node representations in LEHD on Uniform CVRP-100.** Values denote mean cosine similarity ($\pm$ std) across different problem instances and decoding steps for current, depot, and random customer node representations. Enc. and Dec. denote encoder and decoder layers.

Layer	Current Node	Depot Node	Random Node
Enc. 1	0.513 ± 0.330	0.514 ± 0.342	$\mathbf{0.514} \pm 0.327$
Dec. 1	$\mathbf{0.671} \pm 0.251$	0.464 ± 0.296	0.279 ± 0.383
Dec. 2	$\mathbf{0.792} \pm 0.176$	0.502 ± 0.279	0.344 ± 0.331
Dec. 3	$\mathbf{0.772} \pm 0.146$	0.684 ± 0.155	0.395 ± 0.298
Dec. 4	0.764 ± 0.135	$\mathbf{0.774} \pm 0.113$	0.401 ± 0.307
Dec. 5	0.715 ± 0.145	$\mathbf{0.755} \pm 0.135$	0.431 ± 0.263
Dec. 6	$\mathbf{0.735} \pm 0.213$	0.721 ± 0.234	0.732 ± 0.209

G EXPERIMENTAL SETTINGS

This section summarizes the implementation details and hyperparameter settings of the experiments. All models were evaluated using greedy decoding without model-specific test-time augmentation. All experiments were conducted on a single NVIDIA GeForce RTX 3090 GPU.

G.1 ONION-DEPTH ANALYSIS

For the onion-depth analysis, we set the normalized depth threshold to $\tau = 0.5$. For an onion decomposition discretized into B depth bins, the minimum depth required for an oscillation is

$$b_{\min} = \min\left(B, \lfloor 0.5B \rfloor + 1\right). \tag{4}$$

An oscillation is counted when the compressed bin sequence leaves the outermost bin, reaches at least $b_{\min}$, and subsequently returns to the outermost bin. Consecutive repetitions of the same bin are removed before counting.

G.2 TARGETED START-NODE INTERVENTION

For the targeted start-node intervention, donor nodes were selected from the already visited portion of the partial tour. Let S denote the original start node, C the current node, and d a candidate donor node. We define

$$\theta_d = \angle\left(x_S - x_C,\, x_d - x_C\right), \qquad r_d = \frac{\|x_d - x_C\|_2}{\|x_S - x_C\|_2}. \tag{5}$$

The geometric constraints used for the three donor-selection conditions are reported in Table 23.

Table 23: Geometric constraints used for donor selection in the targeted start-node intervention.

Donor condition	Angular constraint	Distance constraint
Maximum angle without distance constraints	Maximize θ_d	None
Maximum angle under similar distance	Maximize θ_d	$0.80 \le r_d \le 1.20$
Similar angle with different distance	$\theta_d \le 15^\circ$	$r_d \le 0.70$ or $r_d \ge 1.30$

Thus, similar distance permits a deviation of at most 20% from the original start-to-current distance. The similar-angle condition restricts the angular deviation to 15°, while requiring the donor distance to be at least 30% smaller or larger.

G.3 FUTURE-ACTION PLANNING PROBES

For each model layer and prediction horizon, we trained a separate linear candidate-ranking probe while keeping the underlying routing model frozen. The data-split and optimization settings are summarized in Table 24.

Table 24: Training settings for the future-action planning probes.

Setting	Value
Training/validation/test split	$70\%/15\%/15\%$
Random seed	123
Training epochs	3
Learning rate	10^{-3}
Weight decay	10^{-4}
Gradient clipping norm	1.0
Fallback batch size	16

G.4 Training Settings

For the standard benchmark experiments, we used the official checkpoints released by the authors to ensure that our baseline results faithfully match the original implementations. To analyze the effect of the training data distribution, we retrained each model on clustered instances while keeping its architecture and original training paradigm unchanged.

We did not switch any model to a different training paradigm (e.g., RL to SL or SL to RL). Such a switch is not straightforward and would change the method under comparison. AM and POMO are originally RL methods: AM is trained with REINFORCE using a rollout baseline, while POMO uses REINFORCE with multiple starting nodes and a shared baseline. These mechanisms are specific to the RL formulation, and adapting them to SL would require a non-trivial redesign of the training objective and targets. For LEHD, the original method is SL; the authors explicitly note that RL training is impractical because the heavy decoder structure incurs substantial memory and computational costs. We therefore restrict our experiments to each model's original training setting, which allows us to isolate the effect of the training distribution rather than confounding it with a change in the learning algorithm.