# RETHINKING ORDER-FIRST SPLIT-SECOND FOR VEHICLE ROUTING PROBLEMS IN NEURAL COMBINATORIAL OPTIMIZATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Neural combinatorial optimization (NCO) has achieved promising performance in vehicle routing problems (VRPs), but how solution representation affects the learning process remains unclear. Inspired by the classical order-first, split-second strategy, we investigate whether neural routing models can benefit from generating only customer orders while delegating route partitioning to an exact Split decoder. We conduct a systematic comparison between direct route construction and customer-ordering policies with Split decoding under single-task learning, multi-task learning, and various generalization settings. Experiments are performed on synthetic instances, public benchmarks, larger problem sizes, and unseen combinations of routing constraints. The results show that replacing route construction with customer ordering does not provide consistent improvements. Although Split can always obtain the best feasible partition for a given customer order, it cannot modify an inappropriate order or recover information unavailable to the policy. The performance gap becomes more significant when constraints, such as hard time windows, strongly couple customer ordering and route planning. These findings indicate that reducing the number of neural decisions alone is insufficient to improve routing performance. Effective neural routing methods require joint consideration of solution representation, available information, and downstream optimization procedures.

## 1 INTRODUCTION

Vehicle routing problems (VRPs) are fundamental combinatorial optimization problems with broad applications in logistics and transportation (Kool et al., 2019; Li et al., 2021; Xin et al., 2021). A routing solution must simultaneously determine customer visiting sequences, vehicle assignments, and feasible routes while satisfying various constraints, including capacity limitations, route duration, service requirements, and time windows. Classical approaches explicitly model these structural characteristics through mathematical programming, local search, and specialized solution representations (Solomon, 1987; Prins, 2004; Vidal et al., 2012; 2013; 2014; Vidal, 2016; Santana et al., 2023; Vidal, 2022).

Neural combinatorial optimization (NCO) provides an alternative paradigm by learning routing policies from distributions of problem instances, avoiding the need for manually designed heuristics or labeled optimal solutions (Vinyals et al., 2015; Bello et al., 2017; Bengio et al., 2021; Cappart et al., 2023; Meng et al., 2025). Recent neural routing methods, such as the Attention Model (AM) and POMO, have achieved competitive performance on representative routing problems (Kool et al., 2019; Kwon et al., 2020). Meanwhile, multi-task learning approaches and generalization-oriented methods have extended neural solvers toward diverse routing constraints and unseen problem distributions (Zhou et al., 2024; Berto et al., 2025; Pan et al., 2025; Joshi et al., 2021; Geisler et al., 2022; Zhou et al., 2023; Huang et al., 2025). Recent studies have also revisited decomposition-based representations in neural routing, such as Cluster-First-Route-Second frameworks (Chin & Schiffer, 2026). However, existing studies primarily focus on model architectures, training strategies, and inference procedures, whereas the role of solution representation in neural routing remains comparatively less explored.



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

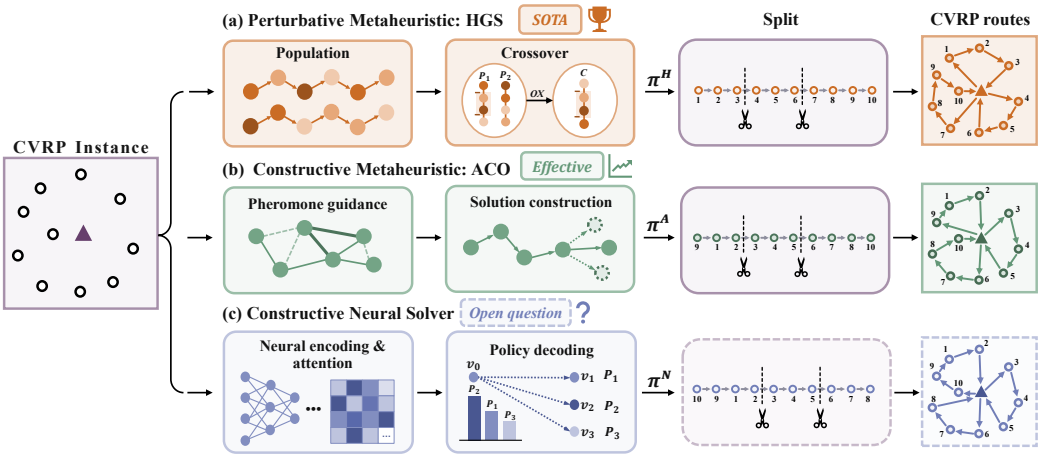



Figure 1: The controlled comparison changes where route-boundary decisions are made and what information the policy observes. Direct policies learn ordering and partitioning jointly; Order–Split policies learn only an order and delegate the best feasible contiguous partition to an exact dynamic program. Both are scored by total distance and checked by strict route replay.

Most neural construction methods follow a direct route construction paradigm, where the policy jointly determines customer ordering and route boundaries. Through dynamic states and feasibility masks, these methods can adapt decisions according to current constraints, such as vehicle capacity and time windows. However, this formulation also requires the neural policy to learn both customer ordering and route partitioning, raising a fundamental question: can part of the routing decision process be effectively delegated to an exact optimization procedure?

The classical order-first split-second (OFSS) scheme provides a natural perspective for addressing this question. Instead of directly constructing complete routes, this framework first generates a giant tour representing the customer order and then applies the Split algorithm to partition the order into feasible vehicle routes (Prins et al., 2014). Given a fixed customer sequence, Split obtains the minimum-cost feasible contiguous partition through dynamic programming (Prins, 2004). Perturbative evolutionary algorithms such as hybrid genetic search (HGS) and its variants have maintained dominance for VRPs utilizing OFSS (Vidal, 2016; 2022). This fact aligns with intuition. For a mechanism that generates solutions via random perturbations like crossover and mutation, OFSS can effectively reduce the search space. Interestingly, even constructive metaheuristics like ant colony optimization (ACO) have demonstrated the effectiveness of OFSS (Jia et al., 2022).

Despite the advantages of OFSS for metaheuristics, whether it benefits neural constructive routing is still unclear. On the one hand, extensive studies in metaheuristics have shown that OFSS is effective by learning customer orders while delegating route partitioning to exact Split algorithms (Vidal, 2016; 2022). On the other hand, recent advances in constructive NCO have demonstrated the effectiveness of end-to-end learning, where neural policies directly learn complete routing strategies from data (Kool et al., 2019; Kwon et al., 2020; Bengio et al., 2021). These two paradigms represent different assumptions about solution learning: decomposing routing decisions with exact algorithms or jointly learning the complete routing structure with neural models. To the best of our knowledge, OFSS has been tried twice in constructive NCO methods (Yaddaden et al., 2022; Wang et al., 2026a). However, existing studies mainly focus on developing hybrid neural solvers rather than isolating the effect of OFSS representation itself. Therefore, whether OFSS can provide consistent benefits to neural constructive routing remains an open question.

In this work, we provide a systematic reassessment of OFSS in neural routing. Different from previous studies that directly adopt this representation, we conduct controlled comparisons with direct route construction under matched neural backbones, training settings, and evaluation protocols. Specifically, we investigate the following three questions:

- **Can OFSS improve single-task routing performance?**

- **Can OFSS support unified multi-task learning with heterogeneous routing constraints?**
- **How does it affect generalization across different problem sizes, distributions, benchmark instances, and unseen routing variants?**

Our experiments cover representative neural routing settings under different learning and generalization scenarios. The results reveal that replacing direct route construction with Split-based route partitioning does not consistently improve routing performance. Although Split guarantees the optimal partition for a given customer order, it cannot compensate for insufficient information during order generation. These findings suggest that effective neural combinatorial optimization requires joint consideration of solution representation, policy inputs, and downstream optimization procedures.

## 2 Related Work

**Neural combinatorial optimization.** Neural construction methods learn a policy that generates a solution directly from an instance, typically in an autoregressive manner. Pointer Networks first demonstrated supervised sequence construction for the TSP (Vinyals et al., 2015), after which reinforcement learning was adopted for the TSP and routing problems (Bello et al., 2017; Nazari et al., 2018). The Attention Model (AM) introduced an encoder–decoder architecture applicable to multiple routing problems (Kool et al., 2019); POMO and Sym-NCO subsequently improved training and inference by exploiting multiple equivalent starts and problem symmetries (Kwon et al., 2020; Kim et al., 2022). These methods reuse static node embeddings across construction steps and are therefore computationally efficient. Their performance, however, can deteriorate under shifts in instance size, distribution, or constraints (Joshi et al., 2021; Geisler et al., 2022; Zhou et al., 2023). Recent solvers allocate more computation to dynamic decoding or strengthen the use of context to improve out-of-distribution generalization (Drakulic et al., 2023; Luo et al., 2023; Fang et al., 2024; Wang et al., 2024; Huang et al., 2025; Xiao et al., 2025), while multi-task models share parameters across VRP variants with different combinations of constraints (Zhou et al., 2024; Li et al., 2025; Berto et al., 2025; Pan et al., 2025; Wang et al., 2026b; Gui et al., 2026; Meng et al., 2026). Rather than proposing another architectural extension or expanding task coverage, we study whether route partitioning should be performed by the neural construction policy or by an exact Split algorithm after the policy has generated a giant tour of customer visiting sequence.

**Split-based methods.** The order-first, split-second (OFSS) framework decomposes a vehicle routing solution into two stages: the first stage determines a permutation representing the customer visiting order, and the second stage partitions this permutation into feasible vehicle routes (Prins et al., 2014). For a fixed giant tour of customer visiting sequence, Split finds the minimum-cost feasible contiguous partition using a dynamic program that finds a shortest path in an acyclic auxiliary graph (Prins, 2004). For the capacity constraint, specialized Split algorithms achieve $O(n)$ time complexity (Vidal, 2016). This representation is not merely a decoding convenience: it is a central component of effective evolutionary and hybrid genetic search methods (Prins, 2004; Vidal et al., 2012; 2013; 2014; Vidal, 2016; 2022; Santana et al., 2023). It has also been studied within ant colony optimization, where direct and order-first, split-second encodings exhibit different strengths across problem settings (Jia et al., 2022). Most closely related to our neural formulation, NOFSS learns a customer-ordering policy and applies Split to construct CVRP routes (Yaddaden et al., 2022). Accordingly, we do not claim that combining a neural customer-ordering policy with Split decoding is new. Instead, we provide a controlled comparison with direct route construction across representative single-task and multi-task neural construction methods and heterogeneous constraint settings. We validate the Split implementations independently and verify every reported solution so that differences in solution quality are not confounded with implementation errors or infeasibility.

## 3 Preliminaries

Following the compact formulations commonly used for neural routing (Zhou et al., 2023; 2024), we describe a family of vehicle routing problems through a common routing objective and a set of constraints. An instance is denoted by $I = (X, \mathcal{P})$, where $X = (x_0, x_1, \ldots, x_n)$ contains the

locations of one depot and $n$ customers, and $\mathcal{P}$ contains the attributes and parameters required by the problem variant. We index the depot by 0 and the customers by $1, \dots, n$. Traveling from node $i$ to node $j$ incurs cost $d_{ij}$; the experiments use Euclidean distance.

A complete solution $\mathcal{R}$ is a set of vehicle routes that serves every customer exactly once. Each route starts at the depot and, except in variants with open routes, returns to the depot. Let $\mathcal{F}(I)$ denote the set of all complete solutions that satisfy the constraints of instance $I$. The routing problem is

$$\mathcal{R}^\star(I) \in \arg\min_{\mathcal{R}\in\mathcal{F}(I)} C(\mathcal{R}), \qquad C(\mathcal{R}) = \sum_{r\in\mathcal{R}} \sum_{(i,j) \text{ consecutive in } r} d_{ij}. \tag{1}$$

Thus, the objective is shared across problem variants, whereas $\mathcal{F}(I)$ specifies which solutions are feasible.

Our single-task experiments consider capacity constraints and hard time windows (Solomon, 1987). In the capacity setting, customer demands and vehicle capacity determine whether a route is feasible. The time-window setting additionally specifies service durations, customer time windows, travel time, and a depot horizon. Early arrival is followed by waiting; service must begin within the corresponding time window, and the vehicle must return to the depot within its horizon. Waiting and service duration affect feasibility but not travel cost. Our multi-task experiments also include open routes, backhaul requirements, and route duration constraints (Zhou et al., 2024). These conditions change $\mathcal{P}$ and $\mathcal{F}(I)$, but not the common definition of a complete solution or the objective in Eq. (1). The constraint transitions used by route construction and Split decoding are specified in Section 4.

# 4 METHOD

## 4.1 OVERVIEW

We compare two solution representations under a common autoregressive framework. Let $I = (X, \mathcal{P})$ denote a problem instance with depot $x_0$, customer set $\mathcal{N} = \{1, \dots, n\}$, and constraint attributes $\mathcal{P}$. A parameterized policy $p_\theta$ sequentially selects actions from a step-dependent admissible set $\mathcal{A}_t$, yielding an action sequence $\boldsymbol{a} = (a_1, \dots, a_T)$:

$$p_\theta(\boldsymbol{a} \mid I) = \prod_{t=1}^{T} p_\theta\big(a_t \mid a_{<t},\, I\big). \tag{2}$$

A deterministic mapping $\Phi$ converts $\boldsymbol{a}$ into a feasible solution $\mathcal{R} = \Phi(\boldsymbol{a};\, I) \in \mathcal{F}(I)$. The two representations differ in $\mathcal{A}_t$, $T$, and $\Phi$.

In *direct route construction*, the admissible set is $\mathcal{A}_t^{\mathrm{dir}} = \mathcal{U}_t \cup \{0\}$, where $\mathcal{U}_t$ contains the unvisited constraint-feasible customers and 0 represents the depot. Selecting a customer extends the current route; selecting the depot closes it and dispatches a new vehicle. A feasibility mask enforces active constraints at every step. The mapping $\Phi^{\mathrm{dir}}$ segments the action sequence at depot returns:

$$\boldsymbol{a} \sim p_\theta^{\mathrm{dir}}(\cdot \mid I), \qquad \mathcal{R}^{\mathrm{dir}} = \mathrm{routes}(\boldsymbol{a}). \tag{3}$$

Under this representation the policy jointly determines the customer visiting order and the route boundaries.

In *customer ordering with Split decoding*, the admissible set is $\mathcal{A}_t^{\mathrm{ord}} = \mathcal{U}_t \subset \mathcal{N}$. The depot is excluded, no constraint-based mask is applied, and the sequence length is fixed at $T = n$. The policy produces a permutation $\pi = (\pi_1, \dots, \pi_n)$ of $\mathcal{N}$, and the Split algorithm returns the minimum-cost feasible contiguous partition:

$$\pi \sim p_\theta^{\mathrm{ord}}(\cdot \mid I), \qquad \mathcal{R}^{\mathrm{split}} = \mathrm{Split}_I(\pi). \tag{4}$$

This is the standard order-first, split-second (OFSS) construction (Prins, 2004; Prins et al., 2014). Figure 2 illustrates this procedure.

Both representations share the same encoder input and are evaluated under identical feasibility conditions and the objective in Eq. (1). The comparison therefore isolates whether customer ordering and route partitioning are learned jointly by the policy or whether the latter is delegated to an exact combinatorial procedure.

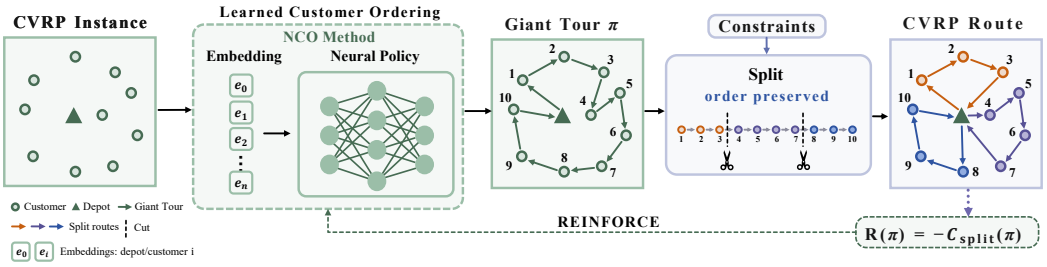



Figure 2: Customer-ordering policy followed by Split decoding. The policy autoregressively constructs customer permutations $\pi$ without constraint-based feasibility masks. For each $\pi$, the Split algorithm receives the full constraint attributes $\mathcal{P}$ and solves the shortest-path recurrence in Eq. (8) to obtain the minimum-cost feasible contiguous partition. During training, the negative objective value $-D(n)$ returned by Split serves as the terminal reward.

### 4.2 Neural Construction Policies

The models construct solutions with autoregressive encoder-decoder architectures. The encoder maps input node features to embeddings $H \in \mathbb{R}^{(n+1)\times d}$. At step $t$, the decoder produces a distribution over candidate actions:

$$p_\theta(a_t \mid a_{<t}, I) = \frac{\exp\big(f_\theta(H, c_t)_{a_t} + m_{t,a_t}\big)}{\sum_{a' \in \mathcal{A}_t} \exp\big(f_\theta(H, c_t)_{a'} + m_{t,a'}\big)}, \tag{5}$$

where $f_\theta$ is the learned compatibility function, $c_t$ the decoding context, and $m_t \in \{0, -\infty\}^{n+1}$ the action mask.

**Direct route construction.** The candidate actions comprise all unvisited customers and the depot. A dynamic state $s_t = (v_t, \tau_t, \ell_t, \dots)$ maintains the current vehicle's position, elapsed time, remaining load, and any additional resource variables. The mask $m_t$ sets inadmissible actions to $-\infty$ based on capacity, time windows, route duration, and other active constraints. The context $c_t$ encodes the current vehicle state and the last visited node, enabling constraint-aware decisions at every step.

**Customer ordering.** The candidate actions contain only unvisited customers. No constraint-based mask is applied: $m_{t,j} = -\infty$ only for previously visited customers. Apart from removing constraint-based feasibility masks, the input information and decoding context of the corresponding direct policy are retained. The resulting policy produces exactly $n$ actions:

$$p_\theta^{\mathrm{ord}}(\pi \mid I) = \prod_{t=1}^{n} p_\theta^{\mathrm{ord}}(\pi_t \mid \pi_{<t},\, I). \tag{6}$$

This formulation applies uniformly across single-task and multi-task settings. All constraint enforcement is delegated to Split decoding. For each backbone, the encoder-decoder architecture and training procedure are preserved; the representation change is confined to the action space and the masking mechanism.

### 4.3 Split Decoding

Given a fixed permutation $\pi = (\pi_1, \dots, \pi_n)$ and instance $I$, Split finds the minimum-cost contiguous partition into feasible routes via a shortest-path computation on a directed acyclic graph.

Define the vertex set $V = \{0, 1, \dots, n\}$, where vertex $j$ represents the state in which customers $\pi_1, \dots, \pi_j$ have been assigned to routes. For each pair $(i, j)$ with $0 \le i < j \le n$, the subsequence $\pi_{i+1:j}$ induces a candidate route starting from the depot, visiting $\pi_{i+1}, \dots, \pi_j$ in order, and returning to the depot for closed-route variants.

Let $\mathcal{C} = \{C_1, \ldots, C_K\}$ denote the active constraints. Each $C_k$ defines a Boolean predicate $F_k$ evaluated incrementally along the route. The composite feasibility indicator is

$$F_I(i,j) = \prod_{k=1}^{K} F_k\big(r_{ij}, \mathcal{P}\big), \tag{7}$$

where $r_{ij}$ denotes the candidate route induced by $\pi_{i+1:j}$. For each feasible arc, the edge weight $c_I(i,j)$ equals the travel cost of the corresponding route.

Let $D(j)$ denote the minimum total cost to serve the first $j$ customers in $\pi$. The optimal partition satisfies the recurrence

$$D(0) = 0, \qquad D(j) = \min_{\substack{0 \le i < j \\ F_I(i,j)=1}} \big\{D(i) + c_I(i,j)\big\}, \quad j = 1, \ldots, n. \tag{8}$$

A predecessor function $P(j)$ records the optimal split point at each vertex. Route boundaries are recovered by backtracking from $j = n$ to 0. The resulting solution $\mathrm{Split}_I(\pi)$ is the minimum-cost feasible contiguous partition of $\pi$.

The general recurrence runs in $O(n^2)$ time. Under the capacity constraint alone, the complexity reduces to $O(n)$ (Vidal, 2016). When time windows or duration limits are active, the worst case remains $O(n^2)$. For a given $\pi$, Split returns the globally optimal contiguous partition. However, it cannot reorder customers and therefore cannot compensate for a poor ordering produced by the policy.

### 4.4 Training Objective

Both representations are trained via REINFORCE to minimize the expected solution cost. For direct construction the cost is $C^{\mathrm{dir}}(\boldsymbol{a}; I) = C\big(\mathrm{routes}(\boldsymbol{a})\big)$; for customer ordering it is $C^{\mathrm{split}}(\pi; I) = C\big(\mathrm{Split}_I(\pi)\big) = D(n)$. The respective objectives are

$$\mathcal{L}^{\mathrm{dir}}(\theta) = \mathbb{E}_{I\sim\mathcal{D}}\, \mathbb{E}_{\boldsymbol{a}\sim p_\theta^{\mathrm{dir}}}\big[C^{\mathrm{dir}}(\boldsymbol{a}; I)\big], \tag{9}$$

$$\mathcal{L}^{\mathrm{split}}(\theta) = \mathbb{E}_{I\sim\mathcal{D}}\, \mathbb{E}_{\pi\sim p_\theta^{\mathrm{ord}}}\big[C^{\mathrm{split}}(\pi; I)\big], \tag{10}$$

where $\mathcal{D}$ is the training distribution. Since Split is non-differentiable, the gradient of $\mathcal{L}^{\mathrm{split}}$ is estimated via the log-derivative trick:

$$\nabla_\theta \mathcal{L}^{\mathrm{split}} = \mathbb{E}_{I,\pi}\Big[\big(C^{\mathrm{split}}(\pi; I) - b(I)\big)\, \nabla_\theta \log p_\theta^{\mathrm{ord}}(\pi \mid I)\Big], \tag{11}$$

where $b(I)$ is a baseline conditionally independent of $\pi$ given $I$ (Williams, 1992). The direct construction gradient takes the same form with $C^{\mathrm{dir}}$ and $\log p_\theta^{\mathrm{dir}}$ replacing their Split counterparts. AM variants employ a greedy rollout baseline; POMO and POMO-based multi-task models use the within-instance mean cost across multiple starting points. The POMO estimator with its sample-dependent within-instance baseline is given in Appendix A.1. MVMoE variants additionally retain their auxiliary load-balancing loss (Zhou et al., 2024).

## 5 Experiments

We evaluate three questions: whether a customer-ordering policy followed by Split decoding improves upon direct route construction under matched settings; whether the same representation remains effective in multi-task learning across training and unseen VRP variants; and whether the conclusions persist at unseen problem sizes and on public benchmarks.

The single-task experiments use AM (Kool et al., 2019) and POMO (Kwon et al., 2020) on CVRP and CVRPTW. The multi-task experiments use POMO-MTL, MVMoE/4E, and MVMoE/4E-L under the official MVMoE protocol (Zhou et al., 2024). We additionally evaluate size generalization and performance on CVRPLIB XML100, the CVRPLIB X-set, and Solomon instances. Training settings, test instances, inference procedures, evaluation measures are described in Appendix A.

Table 1: Average objective values on the single-task test sets.

| Problem | Backbone | $n$ | Direct | Split | Gap |
|---|---|---|---|---|---|
| CVRP | AM | 50 | 10.6905 | 11.1037 | $+3.87\%$ |
| | POMO | 50 | 10.4558 | 10.6989 | $+2.32\%$ |
| | AM | 100 | 16.3084 | 17.2217 | $+5.60\%$ |
| | POMO | 100 | 15.8511 | 16.5395 | $+4.34\%$ |
| CVRPTW | AM | 50 | 16.7427 | 18.0917 | $+8.06\%$ |
| | POMO | 50 | 15.3269 | 16.3204 | $+6.48\%$ |
| | AM | 100 | 28.4429 | 31.1180 | $+9.41\%$ |
| | POMO | 100 | 25.4150 | 27.9364 | $+9.92\%$ |

### 5.1 Direct Construction Achieves Better Single-Task Solution Quality

Table 1 shows that Direct achieves a lower average objective value in all eight comparisons. Split increases the average objective value by $2.32\%$–$5.60\%$ on CVRP and by $6.48\%$–$9.92\%$ on the DAR CVRPTW instances. For both backbones and both problems, the relative difference is larger at $n = 100$ than at $n = 50$. The difference is also consistently larger on CVRPTW than on CVRP. This pattern is consistent with the stronger dependence between customer order and route quality when temporal feasibility involves travel time, waiting time, service duration, and customer time windows.

The paired comparisons support the same conclusion. On CVRP, Direct obtains a lower objective value on 9,597/9,589 of the 10,000 instances at $n = 50$ and 9,987/9,995 at $n = 100$ for AM/POMO, respectively. The corresponding counts on DAR CVRPTW are 879/944 of 1,000 instances at $n = 50$ and 97/100 of 100 instances at $n = 100$. Across the eight comparisons, the proportion of instances on which Direct performs better therefore ranges from $87.90\%$ to $100\%$. The differences in average objective value are thus observed across a large proportion of the test instances rather than being caused by a small number of unusually difficult cases. All 4,400 solutions evaluated on the DAR instances satisfy the capacity and hard time-window constraints, so the comparison reflects solution quality rather than differences in feasibility.

In this controlled comparison, the Split policies receive the same static instance attributes and dynamic decoder context as their Direct counterparts. The main differences concern the action set and feasibility masks. Direct selects customers and depot visits and applies the capacity and time-window feasibility masks during solution construction. Split selects only unvisited customers and does not use these masks when generating the customer permutation. The complete permutation is subsequently partitioned into feasible routes by Split. Although Split finds a minimum-cost feasible contiguous partition for the given customer order, it cannot change the order itself. The results provide no evidence that removing route-boundary decisions improves solution quality in these settings. Instead, the resulting objective values are consistently higher, particularly on the instances with hard time windows.

We further evaluate these models on the CVRPLIB X-set, CVRPLIB XML100 and Solomon public benchmarks. The aggregate X-set, XML100, and Solomon results are reported in Tables 5,7 and 8, respectively. Per-instance results and representative route visualizations are provided in Appendix D.

### 5.2 Multi-Task Feasibility Does Not Imply Better Solution Quality

Tables 2 and 4, the latter reported in Appendix C, present each VRP variant separately rather than averaging objective values across heterogeneous tasks. On the six variants represented during training, the published direct construction model obtains a lower average objective value in all 36 matched model–task–size comparisons. Relative to the corresponding direct model, the increase for Split ranges from $0.38\%$ on OVRP with 50 customers to $7.96\%$ on VRPTW with 100 customers. The same ordering holds in all 36 comparisons on the six retained unseen variants, where the increase ranges from $0.32\%$ on OVRPL with 50 customers to $7.78\%$ on VRPLTW with 100 customers. The largest differences occur on variants with hard time windows, for which the predicted customer order must permit both temporal feasibility and short routes after partitioning. Although the three customer-ordering models differ in their multi-task architectures, none reverses the ordering relative

to its corresponding direct construction model. Because the direct results are published aggregate means rather than paired per-instance observations, we do not report paired statistical tests for these comparisons.

Table 2: Results on 1,000 test instances of the six VRP variants represented during multi-task training. The task-wise layout and reference values follow MVMoE (Zhou et al., 2024).

| | $n = 50$ | | $n = 100$ | | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|---|
| Method | Obj. | Gap | Obj. | Gap | Method | Obj. | Gap | Obj. | Gap |
| CVRP | | | | | VRPTW | | | | |
| Best solver | 10.334 | 0.000% | 15.504 | 0.000% | Best solver | 14.509 | 0.000% | 24.339 | 0.000% |
| POMO-MTL | 10.437 | 0.987% | 15.790 | 1.846% | POMO-MTL | 15.032 | 3.637% | 25.610 | 5.313% |
| POMO-MTL-Split | 10.713 | 3.669% | 16.593 | 7.021% | POMO-MTL-Split | 15.693 | 8.159% | 27.634 | 13.540% |
| MVMoE/4E | 10.428 | 0.896% | 15.760 | 1.653% | MVMoE/4E | 14.999 | 3.410% | 25.512 | 4.903% |
| MVMoE/4E-Split | 10.674 | 3.288% | 16.470 | 6.231% | MVMoE/4E-Split | 15.621 | 7.663% | 27.469 | 12.860% |
| MVMoE/4E-L | 10.434 | 0.955% | 15.771 | 1.728% | MVMoE/4E-L | 15.013 | 3.500% | 25.519 | 4.927% |
| MVMoE/4E-L-Split | 10.689 | 3.431% | 16.491 | 6.367% | MVMoE/4E-L-Split | 15.624 | 7.682% | 27.549 | 13.190% |
| OVRP | | | | | VRPL | | | | |
| Best solver | 6.498 | 0.000% | 9.828 | 0.000% | Best solver | 10.495 | 0.000% | 15.771 | 0.000% |
| POMO-MTL | 6.671 | 2.634% | 10.169 | 3.458% | POMO-MTL | 10.513 | 0.201% | 15.846 | 0.479% |
| POMO-MTL-Split | 6.696 | 3.048% | 10.358 | 5.397% | POMO-MTL-Split | 10.782 | 2.732% | 16.645 | 5.539% |
| MVMoE/4E | 6.655 | 2.402% | 10.138 | 3.136% | MVMoE/4E | 10.501 | 0.092% | 15.812 | 0.261% |
| MVMoE/4E-Split | 6.696 | 3.041% | 10.320 | 5.009% | MVMoE/4E-Split | 10.740 | 2.338% | 16.526 | 4.786% |
| MVMoE/4E-L | 6.665 | 2.548% | 10.145 | 3.214% | MVMoE/4E-L | 10.506 | 0.131% | 15.821 | 0.323% |
| MVMoE/4E-L-Split | 6.700 | 3.105% | 10.313 | 4.935% | MVMoE/4E-L-Split | 10.753 | 2.457% | 16.546 | 4.912% |
| VRPB | | | | | OVRPTW | | | | |
| Best solver | 8.046 | 0.000% | 11.878 | 0.000% | Best solver | 8.683 | 0.000% | 14.380 | 0.000% |
| POMO-MTL | 8.182 | 1.684% | 12.072 | 1.674% | POMO-MTL | 8.987 | 3.470% | 15.008 | 4.411% |
| POMO-MTL-Split | 8.382 | 4.174% | 12.719 | 7.077% | POMO-MTL-Split | 9.245 | 6.469% | 15.771 | 9.672% |
| MVMoE/4E | 8.170 | 1.540% | 12.027 | 1.285% | MVMoE/4E | 8.964 | 3.210% | 14.927 | 3.852% |
| MVMoE/4E-Split | 8.346 | 3.731% | 12.596 | 6.045% | MVMoE/4E-Split | 9.198 | 5.936% | 15.653 | 8.851% |
| MVMoE/4E-L | 8.176 | 1.605% | 12.036 | 1.368% | MVMoE/4E-L | 8.974 | 3.322% | 14.940 | 3.941% |
| MVMoE/4E-L-Split | 8.362 | 3.932% | 12.613 | 6.188% | MVMoE/4E-L-Split | 9.216 | 6.137% | 15.710 | 9.247% |

We omit objective values for VRPBTW, OVRPBTW, VRPBLTW, and OVRPBLTW because most instances do not receive a feasible solution. Across the three models, only $21.7\%$–$24.5\%$ of the 50-customer instances and $1.6\%$–$3.3\%$ of the 100-customer instances are solved feasibly. For the remaining instances, none of the decoded customer orders admits a feasible contiguous partition. Thus, the backhaul state and feasibility mask used during construction do not ensure that the resulting order is compatible with hard time windows after Split decoding. This is a limitation of the present customer-ordering formulation.

For each of the twelve variants retained in the quantitative comparison, all three customer-ordering models return feasible solutions for every test instance, but their average objective values remain higher than those of the corresponding direct construction models. Feasibility also fails to extend reliably to unseen variants that combine backhauls with hard time windows. Multi-task learning therefore enables one policy to address several VRP variants, but it does not eliminate the dependence of solution feasibility and quality on the customer order supplied to Split.

### 5.3 Direct Construction Generalizes Better to Larger Problem Sizes

**Generalization across problem sizes.** Table 3 evaluates the models trained on instances with 100 customers directly on larger instances, without additional training. Direct route construction obtains a lower average objective value in all six comparisons. For AM, Split increases the average objective value by $6.39\%$ at 200 customers, $17.05\%$ at 500 customers, and $8.98\%$ at 1,000 customers. For POMO, the difference increases steadily with problem size, from $8.65\%$ at 200 customers to $10.23\%$ at 500 customers and $14.35\%$ at 1,000 customers.

The per-instance comparisons show the same pattern: Direct obtains a lower objective value in 5,872 of the 6,000 comparisons across both backbones and all three test sizes. All 12,000 solutions are feasible, so the differences concern solution quality rather than capacity feasibility.

The number of vehicles and the total distance do not always change in the same direction. In particular, AM-Split uses fewer vehicles at 1,000 customers but still produces an average distance

Table 3: Size generalization of models trained on 100-customer instances and evaluated without additional training on 1,000 fixed CVRP instances at each test size. Each Direct and Split entry reports mean objective value/mean number of vehicles.

| Backbone | Test size | Direct | Split | Gap |
|---|---|---|---|---|
| AM | 200 | 29.892/20.93 | 31.802/22.31 | $+6.39\%$ |
| AM | 500 | 69.958/52.55 | 81.888/55.54 | $+17.05\%$ |
| AM | 1,000 | 150.811/130.24 | 164.347/112.79 | $+8.98\%$ |
| POMO | 200 | 28.850/20.77 | 31.344/22.10 | $+8.65\%$ |
| POMO | 500 | 67.401/51.13 | 74.297/54.71 | $+10.23\%$ |
| POMO | 1,000 | 131.945/103.98 | 150.881/109.92 | $+14.35\%$ |

that is $8.98\%$ higher. This is consistent with the stated objective, which minimizes total travel distance rather than the number of vehicles. Overall, the results provide no evidence that removing route-boundary decisions improves generalization to problem sizes larger than those encountered during training. Exact Split decoding finds the best feasible partition of the predicted customer order, but it cannot correct the order itself when the effect of ordering errors becomes more pronounced on larger instances.

## 6 Conclusion

This study examined whether a neural construction method benefits from generating only a customer order and delegating route partitioning to an exact Split algorithm. We considered this question in single-task learning, multi-task learning, and generalization settings, using representative neural construction methods and independently verifying the feasibility and objective value of every reported complete solution. **Overall, the results show that order-first, split-second (OFSS) does not provide an effective solution representation for neural combinatorial optimization under the evaluated settings.** Although Split can optimally partition a fixed customer order, delegating route partitioning to an exact algorithm does not compensate for the loss of route-level decision information during neural construction.

In the single-task experiments, customer-ordering policies followed by Split decoding fail to improve solution quality. Compared with direct construction, they increase the average objective value by $2.32\%$–$5.60\%$ on synthetic CVRP instances and increase travel distance by $6.48\%$–$9.92\%$ on generated CVRPTW instances. Independent exhaustive verification confirms that the Split implementations are correct and all evaluated solutions are feasible, indicating that the performance gap mainly originates from inferior customer orders rather than route partitioning errors.

In the multi-task experiments, customer-ordering models can generate feasible solutions for the represented variants and several unseen constraint combinations. However, their objective values remain $0.38\%$–$7.96\%$ higher on training variants and $0.32\%$–$7.78\%$ higher on retained unseen variants than the corresponding direct construction models. Moreover, the failure on unseen variants combining backhauls and hard time windows reveals that generating a feasible customer order alone is insufficient when multiple constraints strongly interact.

The generalization results further show that OFSS does not provide consistent benefits across different scales and distributions. Direct construction achieves lower objective values on CVRPLIB XML100, Solomon benchmarks, and most larger-scale evaluations. Although the performance gap becomes smaller at 1,000 customers for some settings, the results do not demonstrate a systematic advantage of customer ordering with Split decoding.

Taken together, these findings clarify the limitation of applying OFSS directly to neural construction optimization. Exact route partitioning can optimize the route boundaries for a given customer order, but it cannot correct an unsuitable order generated by the neural policy. Therefore, reducing the number of neural decisions by replacing route construction with customer ordering is insufficient to improve solution quality, feasibility, or generalization. Neural routing methods should carefully preserve the interaction between customer ordering and route planning rather than assuming that exact downstream partitioning can replace route-level decision learning.

## THE USE OF LARGE LANGUAGE MODELS (LLMS)

We employed LLMs solely for language polishing and assistance with simple coding tasks.

## REFERENCES


Irwan Bello, Hieu Pham, Quoc V. Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. In *International Conference on Learning Representations Workshop Track*, 2017.

Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2): 405–421, 2021. doi: 10.1016/j.ejor.2020.07.063. URL https://doi.org/10.1016/j.ejor.2020.07.063.

Federico Berto, Chuanbo Hua, Nayeli Gast Zepeda, André Hottung, Niels Wouda, Leon Lan, Junyoung Park, Kevin Tierney, and Jinkyoo Park. RouteFinder: Towards foundation models for vehicle routing problems. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=QzGLoaOPiY.

Quentin Cappart, Didier Chételat, Elias B. Khalil, Andrea Lodi, Christopher Morris, and Petar Veličković. Combinatorial optimization and reasoning with graph neural networks. *Journal of Machine Learning Research*, 24(130):1–61, 2023.

Samuel J. K. Chin and Maximilian Schiffer. Neural cluster first, route second: One-shot capacitated vehicle routing via differentiable optimal transport. *arXiv preprint arXiv:2605.09301*, 2026. doi: 10.48550/arXiv.2605.09301. URL https://arxiv.org/abs/2605.09301.

Darko Drakulic, Sofia Michel, Florian Mai, Arnaud Sors, and Jean-Marc Andreoli. BQ-NCO: Bisimulation quotienting for efficient neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 77416–77429, 2023. doi: 10.52202/075280-3385. URL https://doi.org/10.52202/075280-3385.

Han Fang, Zhihao Song, Paul Weng, and Yutong Ban. INViT: A generalizable routing problem solver with invariant nested view transformer. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12973–12992. PMLR, 2024. URL https://proceedings.mlr.press/v235/fang24c.html.

Simon Geisler, Johanna Sommer, Jan Schuchardt, Aleksandar Bojchevski, and Stephan Günnemann. Generalization of neural combinatorial solvers through the lens of adversarial robustness. In *International Conference on Learning Representations*, 2022.

Shuangchun Gui, Suyu Liu, Xuehe Wang, and Zhiguang Cao. Chain-of-context learning: Dynamic constraint understanding for multi-task VRPs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=AhE6aSlz5g.

Ziwei Huang, Jianan Zhou, Zhiguang Cao, and Yixin Xu. Rethinking light decoder-based solvers for vehicle routing problems. In *International Conference on Learning Representations*, 2025.

Ya-Hui Jia, Yi Mei, and Mengjie Zhang. Confidence-based ant colony optimization for capacitated electric vehicle routing problem with comparison of different encoding schemes. *IEEE Transactions on Evolutionary Computation*, 26(6):1394–1408, 2022. doi: 10.1109/TEVC.2022.3144142. URL https://doi.org/10.1109/TEVC.2022.3144142.

Chaitanya K. Joshi, Quentin Cappart, Louis-Martin Rousseau, and Thomas Laurent. Learning TSP requires rethinking generalization. In *27th International Conference on Principles and Practice of Constraint Programming (CP 2021)*, volume 210 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 33:1–33:21. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2021. doi: 10.4230/LIPIcs.CP.2021.33. URL https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CP.2021.33.

Minsu Kim, Junyoung Park, and Jinkyoo Park. Sym-NCO: Leveraging symmetricity for neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 35, pp. 1936–1949, 2022.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019.

Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. POMO: Policy optimization with multiple optima for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21188–21198, 2020.

Han Li, Fei Liu, Zhi Zheng, Yu Zhang, and Zhenkun Wang. CaDA: Cross-problem routing solver with constraint-aware dual-attention. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 35438–35456. PMLR, 2025. URL https://proceedings.mlr.press/v267/li25bi.html.

Sirui Li, Zhongxia Yan, and Cathy Wu. Learning to delegate for large-scale vehicle routing. In *Advances in Neural Information Processing Systems*, volume 34, pp. 26198–26211, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/dc9fa5f217a1e57b8a6adeb065560b38-Abstract.html.

Fu Luo, Xi Lin, Fei Liu, Qingfu Zhang, and Zhenkun Wang. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8845–8864, 2023.

Dian Meng, Zhiguang Cao, Jie Gao, Yaoxin Wu, and Yaqing Hou. UniteFormer: Unifying node and edge modalities in transformers for vehicle routing problems. In *Advances in Neural Information Processing Systems*, volume 38, pp. 104129–104153, 2025. doi: 10.52202/085713-3139. URL https://doi.org/10.52202/085713-3139.

Dian Meng, Yaoxin Wu, Yaqing Hou, and Zhiguang Cao. PoMtVRS: Preference-optimized multi-task vehicle routing solver with preference gating. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/60922.

MohammadReza Nazari, Afshin Oroojlooy, Lawrence V. Snyder, and Martin Takáč. Reinforcement learning for solving the vehicle routing problem. In *Advances in Neural Information Processing Systems*, volume 31, pp. 9839–9849, 2018.

Yuxin Pan, Zhiguang Cao, Chengyang Gu, Liu Liu, Peilin Zhao, Yize Chen, and Fangzhen Lin. Multi-task vehicle routing solver via mixture of specialized experts under state-decomposable MDP. In *Advances in Neural Information Processing Systems*, volume 38, pp. 36668–36711, 2025. doi: 10.52202/085713-1093. URL https://doi.org/10.52202/085713-1093.

Christian Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002, 2004. doi: 10.1016/S0305-0548(03)00158-8.

Christian Prins, Philippe Lacomme, and Caroline Prodhon. Order-first split-second methods for vehicle routing problems: A review. *Transportation Research Part C: Emerging Technologies*, 40:179–200, 2014. doi: 10.1016/j.trc.2014.01.011.

Ítalo Santana, Andrea Lodi, and Thibaut Vidal. Neural networks for local search and crossover in vehicle routing: A possible overkill? In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pp. 184–199. Springer, 2023. doi: 10.1007/978-3-031-33271-5_13.

Marius M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265, 1987. doi: 10.1287/opre.35.2.254. URL https://doi.org/10.1287/opre.35.2.254.

Thibaut Vidal. Technical note: Split algorithm in $O(n)$ for the capacitated vehicle routing problem. *Computers & Operations Research*, 69:40–47, 2016. doi: 10.1016/j.cor.2015.11.012. URL https://doi.org/10.1016/j.cor.2015.11.012.

Thibaut Vidal. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Computers & Operations Research*, 140:105643, 2022. doi: 10.1016/j.cor.2021.105643. URL https://doi.org/10.1016/j.cor.2021.105643.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, Nadia Lahrichi, and Walter Rei. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60 (3):611–624, 2012. doi: 10.1287/opre.1120.1048.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time windows. *Computers & Operations Research*, 40(1):475–489, 2013. doi: 10.1016/j.cor.2012.07.018.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research*, 234(3):658–673, 2014. doi: 10.1016/j.ejor.2013.09.045.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28, pp. 2692–2700, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/29921001f2f04bd3baee84a12e98098f-Abstract.html.

Chenguang Wang, Zhouliang Yu, Stephen McAleer, Tianshu Yu, and Yaodong Yang. ASP: Learn a universal neural solver! *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46 (6):4102–4114, 2024. doi: 10.1109/TPAMI.2024.3352096. URL https://doi.org/10.1109/TPAMI.2024.3352096.

Wen Wang, Xiangchen Wu, Liang Wang, Hao Hu, and Xianping Tao. A unified knowledge embedded reinforcement learning-based framework for generalized capacitated vehicle routing problems. In *Proceedings of the Thirty-Fifth International Joint Conference on Artificial Intelligence, IJCAI-26*, pp. 6208–6216. International Joint Conferences on Artificial Intelligence Organization, 2026a. doi: 10.24963/ijcai.2026/691. URL https://doi.org/10.24963/ijcai.2026/691.

Yang Wang, Ya-Hui Jia, Wei-Neng Chen, and Yi Mei. Distance-aware attention reshaping for enhancing generalization of neural solvers. *IEEE Transactions on Neural Networks and Learning Systems*, 36(10):18900–18914, 2025. doi: 10.1109/TNNLS.2025.3588209. URL https://doi.org/10.1109/TNNLS.2025.3588209.

Yang Wang, Ya-Hui Jia, Wei-Neng Chen, and Yi Mei. Soft parameter sharing model for cross-problem generalization in vehicle routing problems. *IEEE Transactions on Artificial Intelligence*, 7(1):471–485, 2026b. doi: 10.1109/TAI.2025.3576336.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Yubin Xiao, Yuesong Wu, Rui Cao, Di Wang, Zhiguang Cao, Xuan Wu, Peng Zhao, Yuanshu Li, You Zhou, and Yuan Jiang. DGL: Dynamic global-local information aggregation for scalable VRP generalization with self-improvement learning. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 8669–8677. International Joint Conferences on Artificial Intelligence Organization, 2025. doi: 10.24963/ijcai.2025/964. URL https://doi.org/10.24963/ijcai.2025/964.

Liang Xin, Wen Song, Zhiguang Cao, and Jie Zhang. NeuroLKH: Combining deep learning model with Lin–Kernighan–Helsgaun heuristic for solving the traveling salesman problem. In *Advances in Neural Information Processing Systems*, volume 34, pp. 7472–7483, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/3d863b367aa379f71c7afc0c9cdca41d-Abstract.html.

Ali Yaddaden, Sébastien Harispe, and Michel Vasquez. Neural order-first split-second algorithm for the capacitated vehicle routing problem. In *Optimization and Learning*, volume 1684 of *Communications in Computer and Information Science*, pp. 168–185. Springer, 2022. doi: 10.1007/978-3-031-22039-5_14.

Jianan Zhou, Yaoxin Wu, Wen Song, Zhiguang Cao, and Jie Zhang. Towards omni-generalizable neural methods for vehicle routing problems. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 42769–42789. PMLR, 2023. URL https://proceedings.mlr.press/v202/zhou23o.html.

Jianan Zhou, Zhiguang Cao, Yaoxin Wu, Wen Song, Yining Ma, Jie Zhang, and Xu Chi. MV-MoE: Multi-task vehicle routing solver with mixture-of-experts. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61804–61824. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhou24c.html.

# A Implementation Details

## A.1 Training and Controlled Comparison

For the single-task experiments, we independently train a direct route construction policy and a customer-ordering policy followed by Split decoding at $n \in \{50, 100\}$ for both AM (Kool et al., 2019) and POMO (Kwon et al., 2020). AM is trained for 100 epochs and POMO for 2,000 epochs. All reported results use the model parameters obtained at the end of the fixed training budget; no intermediate model is selected according to test performance. We train one model for each setting and therefore do not report variation across random seeds.

The two policies use the same static node features within each problem setting. For CVRP, the customer features contain coordinates and demand. For CVRPTW, they additionally contain service duration and customer time windows. The depot is embedded from its coordinates. The two policies also receive the same applicable dynamic decoder state, including remaining capacity and, for CVRPTW, the current route time.

The direct policy selects either an unvisited customer or the depot. It applies the capacity and time-window feasibility masks and determines the customer order and route boundaries jointly. Selecting the depot closes the current route and resets the corresponding vehicle state.

The customer-ordering policy selects only unvisited customers. Apart from its customer-only action set and the removal of constraint-based feasibility masks, it retains the input information and decoding context of the corresponding direct policy. After all customers have been selected, Split receives the complete instance data and determines a minimum-cost feasible contiguous partition of the fixed customer order.

Both variants are trained using the objective value of the complete solution. For the direct policy, this is the total distance of the routes constructed by the policy. For the customer-ordering policy, it is the total distance returned by Split after partitioning the generated order. Split is not differentiated; its terminal cost is used in the policy-gradient estimator, and gradients are computed only through the log probabilities of the ordering actions.

**POMO within-instance baseline.** For POMO, let $\pi^{(1)}, \ldots, \pi^{(K)}$ denote the customer orders generated from the $K$ starting customers, where $s_k$ is the starting customer of rollout $k$. Define

$$C_k^{\text{split}} = C^{\text{split}}\left(\pi^{(k)}; I\right), \qquad \overline{C}^{\text{split}} = \frac{1}{K}\sum_{k=1}^{K} C_k^{\text{split}}.$$

The policy-gradient estimator used in training is

$$\widehat{\nabla_\theta \mathcal{L}}_{\text{POMO}}^{\text{split}} = \frac{1}{K}\sum_{k=1}^{K}\left(C_k^{\text{split}} - \overline{C}^{\text{split}}\right)\nabla_\theta \log p_\theta^{\text{ord}}\left(\pi^{(k)} \mid I, s_k\right). \tag{12}$$

The implementation equivalently uses rewards $R_k = -C_k^{\text{split}}$ and subtracts their within-instance mean. Because the mean contains the costs of the sampled rollouts, it is sample-dependent and does not satisfy the conditional-independence assumption used for $b(I)$ in Eq. equation 11. Equation equation 12 is therefore presented as the estimator implemented by POMO rather than as an independent-baseline identity. The Direct variant uses the same estimator with the direct-construction costs and action probabilities.

For each backbone, problem setting, and problem size, the static input features, dynamic decoder state, embedding dimension, encoder depth, optimizer, training distribution, training budget, candidate budget, and inference procedure are matched. The controlled comparison changes the action set, feasibility masking, and the procedure used to obtain complete routes.

## A.2 Single-Task Test Instances and Inference

**Generated CVRP instances.** The CVRP evaluation uses fixed test sets containing 10,000 generated instances at each problem size. We evaluate instances with 50 and 100 customers. AM uses greedy decoding, while POMO uses one starting point per customer. Both methods use eightfold

geometric augmentation. Within each matched comparison, the Direct and Split variants use the same test instances, decoding rule, augmentation procedure, and candidate budget.

**Generated CVRPTW instances.** The CVRPTW evaluation uses the test instances released with Distance-Aware Attention Reshaping (DAR) (Wang et al., 2025), comprising 1,000 instances with 50 customers and 100 instances with 100 customers. Each released instance is evaluated once. AM uses greedy decoding, whereas POMO uses one starting point per customer. No geometric augmentation is applied. Within each matched comparison, the Direct and Split variants use the same instances and inference settings.

**Generalization across problem sizes.** To evaluate generalization across problem sizes, the CVRP models trained on instances with 100 customers are applied without additional training to fixed test sets with 200, 500, and 1,000 customers. Each test set contains 1,000 instances. AM uses greedy decoding, while POMO uses one starting point per customer. Geometric augmentation is not used in this experiment.

**CVRPLIB XML100.** The XML100 evaluation uses all 10,000 instances. Objective values are computed using the integer `EUC_2D` convention. The gap is computed against the proven optimum of each instance before averaging across the test set. AM uses greedy decoding, while POMO uses 100 starting points. Both methods use eightfold geometric augmentation.

**CVRPLIB X-set.** The X-set evaluation uses all 100 official instances, ranging from 100 to 1,000 customers. The models trained on 100-customer CVRP instances are applied without additional training. AM uses greedy decoding, while POMO uses one starting point per customer. Both methods use eightfold geometric augmentation. Coordinates are divided by 1,000 only when supplied to the neural model. Candidate solutions are selected according to total distance computed using the official integer `EUC_2D` convention. For the Split variants, exact capacity-constrained Split is applied to every generated customer order. No repair or local search is used. The reference solution values are accessed only after candidate generation and selection and are used to compute the reported gaps.

**Solomon benchmark.** The Solomon evaluation uses all 56 instances. Each instance is evaluated using its first 50 customers and using all 100 customers. The instances belong to the C1, C2, R1, R2, RC1, and RC2 groups. AM uses greedy decoding, while POMO uses one starting point per customer. Both methods use eightfold geometric augmentation.

Candidate solutions are selected according to total travel distance. The number of vehicles is reported separately and is not used for candidate selection, and no upper bound on the number of vehicles is imposed. The reported results should therefore not be interpreted according to the standard lexicographic criterion that first minimizes the number of vehicles and then minimizes travel distance.

### A.3 Multi-Task Protocol

For the multi-task experiments, we train customer-ordering versions of POMO-MTL, MVMoE/4E, and MVMoE/4E-L (Zhou et al., 2024) at $n \in \{50, 100\}$. Each model is trained for 5,000 epochs using the model dimensions, optimizer, training schedule, and six training variants of the original implementation. All results use the model parameters obtained at the end of the prescribed training budget. We train one model for each setting and do not report variation across random seeds.

The Direct and Split variants use the same static encoder inputs. The depot is represented by its coordinates, while each customer is represented by its coordinates, normalized demand, and time-window bounds. For variants without time windows, the two time-window features are set to zero. Service duration is not included in the static encoder input. The customer-ordering policy also receives the same four dynamic variables used by the original models: load, current time, route length, and the open-route indicator.

The direct policy selects customers and the depot and applies the feasibility masks of the corresponding VRP variant. The customer-ordering policy retains the same instance information and

decoding context but selects only unvisited customers. In accordance with the factorization studied in this paper, the capacity, open-route, route-duration, and hard time-window feasibility masks are disabled during customer-order generation. The only exception is the backhaul admissibility mask described below.

**Backhaul admissibility.** The backhaul exception follows from the signed-demand transition used in the original MVMoE environments. A linehaul customer has positive demand $q_i > 0$, whereas a backhaul customer has negative demand $q_i < 0$. If a new route starts while at least one linehaul customer remains unserved, the vehicle starts with load $Q$. Once only backhaul customers remain, a new route starts with zero load. After visiting customer $i$, the vehicle load changes according to

$$\ell' = \ell - q_i, \qquad 0 \le \ell' \le Q. \tag{13}$$

Consequently, serving a linehaul customer decreases the load, whereas serving a backhaul customer increases it.

Under this definition, an arbitrary customer permutation need not admit a feasible contiguous partition. For example, consider a permutation whose first customer is a backhaul customer while a linehaul customer remains later in the order. Any route containing the first customer must start with load $Q$, and serving that backhaul customer would increase the load to $Q + |q_i| > Q$. Inserting a route boundary cannot repair this violation because the first route is already required to start with a full load. More generally, the same obstruction occurs whenever the next customer can neither continue the current backhaul-feasible segment nor start a new segment under the MVMoE load initialization rule.

Without the backhaul admissibility mask, the policy can therefore generate customer orders for which Split returns no feasible partition and hence an infinite complete-solution cost. Under the original REINFORCE and POMO training procedures, these non-finite costs do not provide a usable policy-gradient signal. We therefore retain the backhaul mask to keep customer-order generation within the domain for which the Split objective is finite.

For each candidate customer, this mask checks whether the customer can either continue an auxiliary backhaul-feasible segment or start a new one while keeping the signed load within $[0, Q]$. If neither transition is feasible, the customer is masked. The auxiliary segment boundaries are used only to maintain a feasible backhaul load transition during customer-order generation. They are not actions selected by the policy and are not supplied to Split.

After the complete customer order has been generated, Split receives the full instance specification, including capacity, open-route status, signed demands, the route-duration limit, service duration, time windows, and the depot horizon. It discards the auxiliary backhaul segmentation, evaluates all feasible contiguous subsequences, and returns a minimum-cost feasible partition of the fixed customer order. Backhaul feasibility is therefore checked again by Split, which remains free to choose route boundaries different from the auxiliary segmentation.

This exception preserves the intended division between customer ordering and route partitioning. The policy still selects only customers and makes no final route-boundary decisions; Split alone constructs the reported routes. The backhaul mask ensures only compatibility with the signed-demand transition. It does not enforce the other routing constraints or guarantee that the order admits a feasible partition when additional constraints are present. This distinction explains why orders satisfying the backhaul mask can still be infeasible on unseen variants that combine backhauls with hard time windows.

The resulting complete-solution cost is used as the terminal reward. Split is not differentiated, and gradients are computed only through the log probabilities of the customer-ordering policy.

The official evaluation contains 1,000 generated instances for each of sixteen VRP variants. Six variants are represented during training: CVRP, OVRP, VRPB, VRPL, VRPTW, and OVRPTW. The remaining ten variants evaluate zero-shot generalization to unseen combinations of capacity, open routes, backhauls, route duration limits, and hard time windows. Results are reported separately for each variant rather than averaging objective values across heterogeneous tasks.

Inference uses greedy decoding, the admissible starting customers defined by the original POMO environments, eightfold geometric augmentation, and random seed 2024. For variants without backhauls, the procedure uses one starting point per customer. When linehaul customers are present in

a backhaul variant, the starting points are restricted to linehaul customers. A backhaul customer cannot start a feasible route while an unserved linehaul customer remains because the route would start with load $Q$ and the pickup would immediately exceed the vehicle capacity.

All ten unseen variants are evaluated. For OVRPB, OVRPL, VRPBL, VRPLTW, OVRPBL, and OVRPLTW, every Split model returns a feasible solution for every instance. For the four remaining variants, which combine backhauls and hard time windows, most generated customer orders do not admit a feasible contiguous partition. These variants are therefore excluded from comparisons of average objective values rather than reporting averages conditional on successful instances. No constraint relaxation, repair procedure, or local search is applied. These failures do not contradict the role of the retained backhaul mask: the mask preserves feasibility only under the signed-demand transition, whereas the combination with hard time windows is evaluated only after the complete customer order has been generated.

We compare the Split models with the aggregate results published for the corresponding direct route construction models. Because the published results do not contain paired outputs for the individual test instances, the comparison uses differences between aggregate means rather than paired instance-level differences. Results for the six training variants are reported in Table 2; results for the six retained unseen variants are reported in Appendix C, Table 4.

### A.4 Evaluation Measures and Feasibility Verification

Before the main experiments, each Split implementation is validated on small instances by exhaustively enumerating all feasible contiguous partitions of a fixed customer order. For every tested constraint combination, the objective value returned by the dynamic program agrees with the minimum value found by enumeration. Because several partitions may have the same objective value, we verify the feasibility and cost of the recovered partition rather than requiring identical route boundaries.

As an additional consistency check, we remove the depot visits from 1,000 feasible solutions produced by direct route construction while preserving the customer order. Applying Split to each resulting order never produces a higher objective value than the original route boundaries, up to numerical tolerance. These checks verify the implementation and the optimality of Split for a fixed customer order; they do not imply that the customer order is globally optimal.

We report average objective value, feasibility rate, and number of vehicles where applicable. Lower objective values are better. Inference time is not reported because the evaluations were not all conducted under the same hardware and batching configuration.

For the CVRPLIB X-set, the relative gap is computed against the reference cost provided with each instance. For XML100, the relative gap is computed for each instance against its proven optimum and is then averaged over the complete test set. For Solomon, total travel distance and the number of vehicles are reported separately because candidate solutions are selected according to total travel distance rather than lexicographically according to vehicle count and distance.

In the multi-task tables, the objective value is averaged over the 1,000 test instances of each problem variant. The reported gap follows MVMoE (Zhou et al., 2024) and is computed against the best traditional solver reported for the corresponding problem variant and problem size. Comparisons between a Split model and its direct route construction counterpart are computed from their aggregate objective values. A paired comparison is reported only when both methods are evaluated on the same saved instances and their individual results are available.

The reported results are obtained from one completed training run for each model and therefore do not estimate variation across training seeds. Test instances are fixed within each matched comparison. Results over a subset of feasible instances are reported only when explicitly identified as conditional averages; they are not presented as averages over the complete test set.

Every saved solution from the generated single-task tests, size-generalization tests, CVRPLIB X-set, XML100, Solomon, and multi-task evaluations is checked by a verifier implemented independently of both the neural policy and the Split decoder. The verifier recomputes the objective value and checks that every customer is served exactly once, with no missing or duplicate visits. It then applies all constraints required by the corresponding problem variant.

For capacity constraints, the verifier checks the vehicle load throughout each route. For variants with backhauls, it additionally checks the prescribed linehaul and backhaul load transitions and requires all linehaul customers on a route to be visited before its backhaul customers. For a route duration limit, it checks the complete route length, including the return edge for a closed route. For hard time windows, it propagates travel, waiting, and service times, checks that service begins within each customer time window, and verifies return to the depot within its operating horizon for a closed route. For an open route, the final depot edge and the corresponding return condition are omitted.

Continuous distances and times are recomputed using double-precision arithmetic. CVRPLIB objective values follow the integer `EUC_2D` convention where required. Each vehicle route is evaluated independently from the depot departure state specified by the instance. If none of the generated customer orders admits a feasible contiguous partition, the corresponding instance is reported as infeasible. No constraint is relaxed, and no repair procedure or local search is applied.

# B Training Dynamics

Figures 3 and 4 report the epoch-level mean complete-solution cost during CVRP training. For Direct, this is the cost of the routes constructed by the policy. For Split, it is the cost obtained after applying Split to the generated customer order. These costs determine the training reward, but they are not the policy-gradient loss itself. Lower values are better.

The curves are plotted directly from the recorded epoch-level means without smoothing. They are computed on newly generated training batches rather than a fixed validation set and therefore describe training behavior rather than generalization performance. AM and POMO also use different rollout and multi-start procedures, so their curves should be interpreted within each backbone rather than as a hardware- or candidate-budget-matched comparison between backbones.

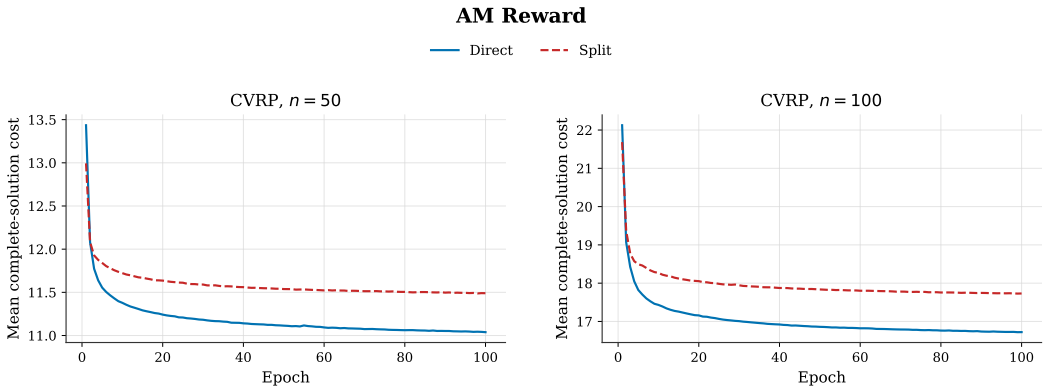



Figure 3: Mean complete-solution cost during training for the Direct and Split variants of AM on CVRP with 50 and 100 customers. The Split cost is evaluated after route partitioning. Curves show the recorded epoch-level means without smoothing. Lower is better.

# C Zero-Shot Generalization Results

Table 4 reports the complete results for the six unseen VRP variants on which all evaluated methods return feasible solutions. The four remaining variants are omitted for the feasibility reasons discussed in Appendix A.3.

# D Benchmark Results

## D.1 CVRPLIB X-set Results

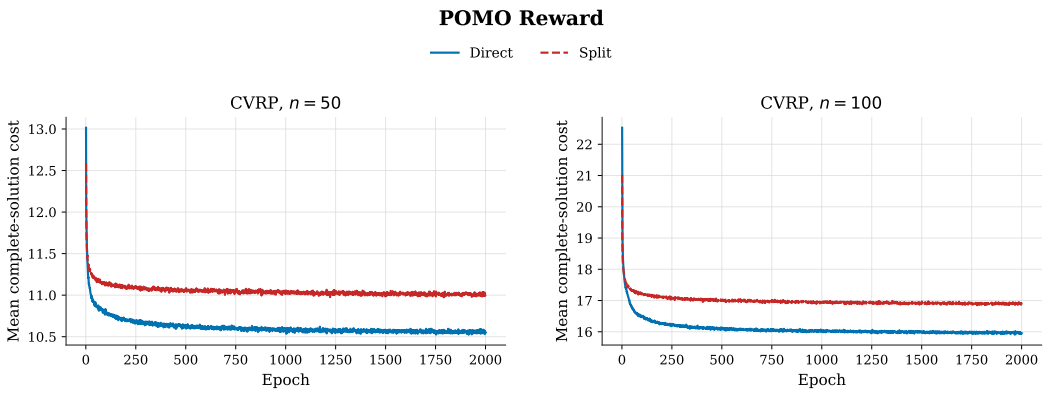



Figure 4: Mean complete-solution cost during training for the Direct and Split variants of POMO on CVRP with 50 and 100 customers. The Split cost is evaluated after route partitioning. Curves show the recorded epoch-level means without smoothing. Lower is better.

Table 4: Zero-shot generalization on 1,000 test instances of six unseen VRP variants, using the task-wise layout and reference values of MVMoE (Zhou et al., 2024). Obj., Gap, result provenance, and omitted timing follow Table 2. Every Split model returns a feasible solution for every instance included here. The four unseen variants combining backhauls and hard time windows are excluded because most instances do not receive a feasible solution. Lower is better.

| | $n = 50$ | | $n = 100$ | | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|---|
| Method | Obj. | Gap | Obj. | Gap | Method | Obj. | Gap | Obj. | Gap |
| OVRPB | | | | | OVRPL | | | | |
| Best solver | 5.745 | 0.000% | 8.365 | 0.000% | Best solver | 6.490 | 0.000% | 9.790 | 0.000% |
| POMO-MTL | 6.116 | 6.430% | 8.979 | 7.335% | POMO-MTL | 6.668 | 2.734% | 10.126 | 3.441% |
| POMO-MTL-Split | 6.314 | 9.900% | 9.570 | 14.408% | POMO-MTL-Split | 6.690 | 3.076% | 10.316 | 5.378% |
| MVMoE/4E | 6.092 | 5.999% | 8.959 | 7.088% | MVMoE/4E | 6.650 | 2.454% | 10.097 | 3.148% |
| MVMoE/4E-Split | 6.300 | 9.669% | 9.481 | 13.336% | MVMoE/4E-Split | 6.690 | 3.088% | 10.271 | 4.916% |
| MVMoE/4E-L | 6.122 | 6.522% | 8.972 | 7.243% | MVMoE/4E-L | 6.659 | 2.597% | 10.106 | 3.244% |
| MVMoE/4E-L-Split | 6.304 | 9.731% | 9.487 | 13.410% | MVMoE/4E-L-Split | 6.693 | 3.130% | 10.271 | 4.909% |
| VRPBL | | | | | VRPLTW | | | | |
| Best solver | 8.029 | 0.000% | 11.790 | 0.000% | Best solver | 14.598 | 0.000% | 25.195 | 0.000% |
| POMO-MTL | 8.188 | 1.971% | 11.998 | 1.793% | POMO-MTL | 14.961 | 2.586% | 25.619 | 1.920% |
| POMO-MTL-Split | 8.399 | 4.607% | 12.652 | 7.314% | POMO-MTL-Split | 15.601 | 6.870% | 27.613 | 9.595% |
| MVMoE/4E | 8.172 | 1.776% | 11.945 | 1.346% | MVMoE/4E | 14.937 | 2.421% | 25.514 | 1.471% |
| MVMoE/4E-Split | 8.365 | 4.181% | 12.510 | 6.104% | MVMoE/4E-Split | 15.534 | 6.411% | 27.442 | 8.917% |
| MVMoE/4E-L | 8.180 | 1.872% | 11.960 | 1.473% | MVMoE/4E-L | 14.953 | 2.535% | 25.529 | 1.545% |
| MVMoE/4E-L-Split | 8.375 | 4.310% | 12.537 | 6.334% | MVMoE/4E-L-Split | 15.559 | 6.581% | 27.501 | 9.152% |
| OVRPBL | | | | | OVRPLTW | | | | |
| Best solver | 5.739 | 0.000% | 8.348 | 0.000% | Best solver | 8.669 | 0.000% | 14.279 | 0.000% |
| POMO-MTL | 6.104 | 6.306% | 8.961 | 7.343% | POMO-MTL | 8.987 | 3.633% | 14.896 | 4.374% |
| POMO-MTL-Split | 6.279 | 9.403% | 9.571 | 14.650% | POMO-MTL-Split | 9.234 | 6.520% | 15.666 | 9.712% |
| MVMoE/4E | 6.076 | 5.843% | 8.942 | 7.115% | MVMoE/4E | 8.966 | 3.396% | 14.828 | 3.903% |
| MVMoE/4E-Split | 6.271 | 9.263% | 9.471 | 13.447% | MVMoE/4E-Split | 9.187 | 5.979% | 15.549 | 8.894% |
| MVMoE/4E-L | 6.104 | 6.310% | 8.957 | 7.300% | MVMoE/4E-L | 8.974 | 3.488% | 14.839 | 3.971% |
| MVMoE/4E-L-Split | 6.274 | 9.327% | 9.493 | 13.720% | MVMoE/4E-L-Split | 9.204 | 6.168% | 15.607 | 9.298% |

Table 6: Per-instance results on all 100 CVRPLIB X-set instances. Ref. is the reference cost from the accompanying solution file. For every method, Cost is the integer `EUC_2D` objective, Gap is relative to Ref., and Veh. is the number of routes. All entries are independently verified.

| Instance | $n$ | Ref. | AM Direct | | | AM Split | | | POMO Direct | | | POMO Split | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Cost | Gap | Veh. | Cost | Gap | Veh. | Cost | Gap | Veh. | Cost | Gap | Veh. |
| X-n101-k25 | 100 | 27,591 | 29,659 | 7.50% | 28 | 30,518 | 10.61% | 30 | 29,443 | 6.71% | 28 | 30,532 | 10.66% | 29 |
| X-n106-k14 | 105 | 26,362 | 27,593 | 4.67% | 14 | 28,485 | 8.05% | 14 | 27,655 | 4.90% | 14 | 27,936 | 5.97% | 14 |
| X-n110-k13 | 109 | 14,971 | 15,815 | 5.64% | 13 | 16,411 | 9.62% | 14 | 15,053 | 0.55% | 13 | 15,674 | 4.70% | 13 |

Continued on next page

Table 6 continued

| Instance | $n$ | Ref. | AM Direct | | | AM Split | | | POMO Direct | | | POMO Split | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Cost | Gap | Veh. | Cost | Gap | Veh. | Cost | Gap | Veh. | Cost | Gap | Veh. |
| X-n115-k10 | 114 | 12,747 | 14,133 | 10.87% | 11 | 14,743 | 15.66% | 11 | 13,362 | 4.82% | 10 | 14,039 | 10.14% | 11 |
| X-n120-k6 | 119 | 13,332 | 14,151 | 6.14% | 6 | 14,809 | 11.08% | 6 | 13,609 | 2.08% | 6 | 14,879 | 11.60% | 7 |
| X-n125-k30 | 124 | 55,539 | 60,147 | 8.30% | 32 | 63,233 | 13.85% | 35 | 59,103 | 6.42% | 32 | 61,890 | 11.44% | 34 |
| X-n129-k18 | 128 | 28,940 | 30,173 | 4.26% | 18 | 31,864 | 10.10% | 19 | 29,788 | 2.93% | 18 | 31,626 | 9.28% | 18 |
| X-n134-k13 | 133 | 10,916 | 12,544 | 14.91% | 15 | 12,647 | 15.86% | 14 | 11,553 | 5.84% | 14 | 12,066 | 10.53% | 14 |
| X-n139-k10 | 138 | 13,590 | 14,232 | 4.72% | 11 | 15,383 | 13.19% | 12 | 14,103 | 3.77% | 11 | 15,516 | 14.17% | 12 |
| X-n143-k7 | 142 | 15,700 | 17,294 | 10.15% | 7 | 17,585 | 12.01% | 7 | 16,140 | 2.80% | 7 | 17,348 | 10.50% | 7 |
| X-n148-k46 | 147 | 43,448 | 51,834 | 19.30% | 53 | 48,920 | 12.59% | 53 | 46,292 | 6.55% | 49 | 48,488 | 11.60% | 51 |
| X-n153-k22 | 152 | 21,220 | 24,577 | 15.82% | 26 | 26,094 | 22.97% | 29 | 23,687 | 11.63% | 26 | 25,909 | 22.10% | 28 |
| X-n157-k13 | 156 | 16,876 | 17,616 | 4.38% | 13 | 18,716 | 10.90% | 13 | 17,691 | 4.83% | 13 | 18,641 | 10.46% | 13 |
| X-n162-k11 | 161 | 14,138 | 15,180 | 7.37% | 11 | 16,514 | 16.81% | 12 | 14,885 | 5.28% | 12 | 16,510 | 16.78% | 12 |
| X-n167-k10 | 166 | 20,557 | 21,671 | 5.42% | 10 | 23,552 | 14.57% | 10 | 21,466 | 4.42% | 10 | 23,430 | 13.98% | 10 |
| X-n172-k51 | 171 | 45,607 | 51,290 | 12.46% | 64 | 51,737 | 13.44% | 60 | 48,420 | 6.17% | 54 | 51,357 | 12.61% | 61 |
| X-n176-k26 | 175 | 47,812 | 53,121 | 11.10% | 29 | 57,586 | 20.44% | 32 | 52,870 | 10.58% | 28 | 58,836 | 23.06% | 33 |
| X-n181-k23 | 180 | 25,569 | 26,559 | 3.87% | 23 | 27,618 | 8.01% | 24 | 26,344 | 3.03% | 23 | 27,488 | 7.51% | 23 |
| X-n186-k15 | 185 | 24,145 | 25,788 | 6.80% | 15 | 27,612 | 14.36% | 16 | 25,306 | 4.81% | 15 | 27,247 | 12.85% | 16 |
| X-n190-k8 | 189 | 16,980 | 19,571 | 15.26% | 8 | 19,843 | 16.86% | 8 | 18,491 | 8.90% | 8 | 19,477 | 14.71% | 8 |
| X-n195-k51 | 194 | 44,225 | 49,620 | 12.20% | 58 | 50,775 | 14.81% | 61 | 47,171 | 6.66% | 54 | 50,729 | 14.71% | 59 |
| X-n200-k36 | 199 | 58,578 | 62,276 | 6.31% | 37 | 65,860 | 12.43% | 39 | 63,997 | 9.25% | 38 | 65,162 | 11.24% | 39 |
| X-n204-k19 | 203 | 19,565 | 20,891 | 6.78% | 19 | 22,296 | 13.96% | 21 | 20,822 | 6.42% | 19 | 22,275 | 13.85% | 21 |
| X-n209-k16 | 208 | 30,656 | 32,749 | 6.83% | 16 | 34,873 | 13.76% | 17 | 31,972 | 4.29% | 16 | 34,122 | 11.31% | 17 |
| X-n214-k11 | 213 | 10,856 | 13,176 | 21.37% | 12 | 13,845 | 27.53% | 13 | 11,831 | 8.98% | 12 | 13,137 | 21.01% | 12 |
| X-n219-k73 | 218 | 117,595 | 128,770 | 9.50% | 77 | 122,696 | 4.34% | 73 | 136,484 | 16.06% | 77 | 123,837 | 5.31% | 73 |
| X-n223-k34 | 222 | 40,437 | 42,954 | 6.22% | 34 | 45,941 | 13.61% | 37 | 42,493 | 5.08% | 35 | 44,938 | 11.13% | 36 |
| X-n228-k23 | 227 | 25,742 | 28,426 | 10.43% | 24 | 31,559 | 22.60% | 27 | 27,915 | 8.44% | 23 | 31,635 | 22.89% | 26 |
| X-n233-k16 | 232 | 19,230 | 21,192 | 10.20% | 17 | 22,124 | 15.05% | 19 | 20,360 | 5.88% | 17 | 22,319 | 16.06% | 18 |
| X-n237-k14 | 236 | 27,042 | 30,638 | 13.30% | 14 | 31,082 | 14.94% | 14 | 28,666 | 6.01% | 14 | 31,482 | 16.42% | 14 |
| X-n242-k48 | 241 | 82,751 | 88,159 | 6.54% | 50 | 93,558 | 13.06% | 53 | 86,302 | 4.29% | 48 | 91,833 | 10.98% | 52 |
| X-n247-k50 | 246 | 37,274 | 43,406 | 16.45% | 57 | 47,410 | 27.19% | 64 | 41,746 | 12.00% | 56 | 46,865 | 25.73% | 62 |
| X-n251-k28 | 250 | 38,684 | 41,161 | 6.40% | 28 | 42,856 | 10.78% | 29 | 40,409 | 4.46% | 28 | 43,056 | 11.30% | 29 |
| X-n256-k16 | 255 | 18,839 | 20,763 | 10.21% | 17 | 21,947 | 16.50% | 17 | 20,591 | 9.30% | 17 | 22,573 | 19.82% | 18 |
| X-n261-k13 | 260 | 26,558 | 29,465 | 10.95% | 13 | 31,732 | 19.48% | 13 | 28,051 | 5.62% | 13 | 31,468 | 18.49% | 14 |
| X-n266-k58 | 265 | 75,478 | 82,733 | 9.61% | 69 | 84,920 | 12.51% | 64 | 85,352 | 13.08% | 67 | 85,787 | 13.66% | 63 |
| X-n270-k35 | 269 | 35,291 | 37,742 | 6.95% | 37 | 39,712 | 12.53% | 40 | 37,219 | 5.46% | 37 | 39,832 | 12.87% | 39 |
| X-n275-k28 | 274 | 21,245 | 25,380 | 19.46% | 31 | 24,036 | 13.14% | 30 | 22,949 | 8.02% | 29 | 24,481 | 15.23% | 30 |
| X-n280-k17 | 279 | 33,503 | 36,382 | 8.59% | 18 | 41,015 | 22.42% | 20 | 35,463 | 5.85% | 17 | 40,932 | 22.17% | 19 |
| X-n284-k15 | 283 | 20,226 | 24,034 | 18.83% | 15 | 24,566 | 21.46% | 16 | 22,541 | 11.45% | 15 | 24,771 | 22.47% | 15 |
| X-n289-k60 | 288 | 95,151 | 102,889 | 8.13% | 64 | 109,551 | 15.13% | 69 | 102,189 | 7.40% | 64 | 108,700 | 14.24% | 68 |
| X-n294-k50 | 293 | 47,161 | 50,865 | 7.85% | 51 | 54,872 | 16.35% | 55 | 50,614 | 7.32% | 52 | 53,841 | 14.16% | 58 |
| X-n298-k31 | 297 | 34,231 | 37,316 | 9.01% | 32 | 40,186 | 17.40% | 35 | 36,850 | 7.65% | 32 | 39,638 | 15.80% | 34 |
| X-n303-k21 | 302 | 21,736 | 23,873 | 9.83% | 21 | 25,900 | 19.16% | 22 | 23,104 | 6.29% | 21 | 26,026 | 19.74% | 22 |
| X-n308-k13 | 307 | 25,859 | 28,790 | 11.33% | 13 | 32,413 | 25.35% | 14 | 27,991 | 8.24% | 13 | 32,131 | 24.25% | 14 |
| X-n313-k71 | 312 | 94,043 | 101,495 | 7.92% | 75 | 108,633 | 15.51% | 82 | 100,452 | 6.81% | 74 | 108,096 | 14.94% | 80 |
| X-n317-k53 | 316 | 78,355 | 82,064 | 4.73% | 54 | 82,593 | 5.41% | 53 | 94,661 | 20.81% | 65 | 83,744 | 6.88% | 54 |
| X-n322-k28 | 321 | 29,834 | 33,062 | 10.82% | 29 | 34,589 | 15.94% | 30 | 33,058 | 10.81% | 29 | 35,257 | 18.18% | 30 |
| X-n327-k20 | 326 | 27,532 | 31,412 | 14.09% | 20 | 32,636 | 18.54% | 21 | 30,157 | 9.53% | 20 | 33,395 | 21.30% | 21 |
| X-n331-k15 | 330 | 31,102 | 35,121 | 12.92% | 15 | 37,104 | 19.30% | 15 | 33,542 | 7.85% | 15 | 37,878 | 21.79% | 15 |
| X-n336-k84 | 335 | 139,111 | 159,916 | 14.96% | 101 | 162,671 | 16.94% | 99 | 151,232 | 8.71% | 91 | 162,422 | 16.76% | 99 |
| X-n344-k43 | 343 | 42,050 | 46,324 | 10.16% | 46 | 48,757 | 15.95% | 47 | 45,235 | 7.57% | 45 | 48,147 | 14.50% | 47 |
| X-n351-k40 | 350 | 25,896 | 29,035 | 12.12% | 41 | 31,060 | 19.94% | 46 | 28,195 | 8.88% | 41 | 31,185 | 20.42% | 45 |
| X-n359-k29 | 358 | 51,505 | 56,461 | 9.62% | 29 | 59,051 | 14.65% | 30 | 54,000 | 4.84% | 29 | 58,793 | 14.15% | 31 |
| X-n367-k17 | 366 | 22,814 | 26,642 | 16.78% | 18 | 29,176 | 27.89% | 18 | 25,002 | 9.59% | 17 | 29,428 | 28.99% | 18 |
| X-n376-k94 | 375 | 147,713 | 155,949 | 5.58% | 96 | 160,610 | 8.73% | 96 | 168,669 | 14.19% | 105 | 159,999 | 8.32% | 95 |
| X-n384-k52 | 383 | 65,940 | 71,094 | 7.82% | 54 | 75,863 | 15.05% | 58 | 70,321 | 6.64% | 54 | 74,659 | 13.22% | 57 |
| X-n393-k38 | 392 | 38,260 | 42,916 | 12.17% | 39 | 44,995 | 17.60% | 41 | 42,215 | 10.34% | 39 | 45,308 | 18.42% | 40 |
| X-n401-k29 | 400 | 66,154 | 72,172 | 9.10% | 29 | 74,089 | 11.99% | 30 | 69,574 | 5.17% | 29 | 74,831 | 13.12% | 30 |
| X-n411-k19 | 410 | 19,712 | 24,755 | 25.58% | 21 | 25,760 | 30.68% | 22 | 22,269 | 12.97% | 19 | 25,753 | 30.65% | 20 |
| X-n420-k130 | 419 | 107,798 | 140,272 | 30.12% | 163 | 133,787 | 24.11% | 156 | 125,583 | 16.50% | 145 | 131,169 | 21.68% | 153 |
| X-n429-k61 | 428 | 65,449 | 71,929 | 9.90% | 63 | 75,679 | 15.63% | 68 | 70,969 | 8.43% | 63 | 75,246 | 14.97% | 67 |
| X-n439-k37 | 438 | 36,391 | 40,797 | 12.11% | 37 | 42,885 | 17.85% | 39 | 42,198 | 15.96% | 40 | 45,095 | 23.92% | 38 |
| X-n449-k29 | 448 | 55,233 | 61,812 | 11.91% | 30 | 65,542 | 18.66% | 31 | 59,592 | 7.89% | 30 | 65,737 | 19.02% | 31 |
| X-n459-k26 | 458 | 24,139 | 29,594 | 22.60% | 27 | 30,394 | 25.91% | 29 | 27,793 | 15.14% | 27 | 31,387 | 30.03% | 29 |
| X-n469-k138 | 468 | 221,824 | 254,982 | 14.95% | 164 | 261,951 | 18.09% | 161 | 260,914 | 17.62% | 157 | 264,828 | 19.39% | 159 |
| X-n480-k70 | 479 | 89,449 | 96,634 | 8.03% | 71 | 103,394 | 15.59% | 76 | 95,916 | 7.23% | 72 | 102,380 | 14.46% | 75 |
| X-n491-k59 | 490 | 66,483 | 72,505 | 9.06% | 60 | 79,741 | 19.94% | 65 | 71,485 | 7.52% | 60 | 78,063 | 17.42% | 65 |
| X-n502-k39 | 501 | 69,226 | 72,829 | 5.20% | 39 | 75,882 | 9.61% | 40 | 77,311 | 11.68% | 40 | 79,672 | 15.09% | 39 |
| X-n513-k21 | 512 | 24,201 | 29,550 | 22.10% | 22 | 30,762 | 27.11% | 23 | 28,190 | 16.48% | 21 | 32,500 | 34.29% | 23 |
| X-n524-k153 | 523 | 154,593 | 218,755 | 41.50% | 210 | 199,727 | 29.20% | 192 | 174,628 | 12.96% | 166 | 195,807 | 26.66% | 187 |
| X-n536-k96 | 535 | 94,846 | 104,881 | 10.58% | 100 | 114,519 | 20.74% | 109 | 113,570 | 19.74% | 106 | 111,444 | 17.50% | 106 |
| X-n548-k50 | 547 | 86,700 | 103,584 | 19.47% | 56 | 97,752 | 12.75% | 51 | 93,506 | 7.85% | 51 | 99,977 | 15.31% | 51 |

Continued on next page

Table 6 continued

| Instance | $n$ | Ref. | AM Direct | | | AM Split | | | POMO Direct | | | POMO Split | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Cost | Gap | Veh. | Cost | Gap | Veh. | Cost | Gap | Veh. | Cost | Gap | Veh. |
| X-n561-k42 | 560 | 42,717 | 48,993 | 14.69% | 43 | 53,211 | 24.57% | 47 | 47,430 | 11.03% | 42 | 53,794 | 25.93% | 47 |
| X-n573-k30 | 572 | 50,673 | 62,460 | 23.26% | 31 | 63,627 | 25.56% | 33 | 58,310 | 15.07% | 30 | 63,446 | 25.21% | 32 |
| X-n586-k159 | 585 | 190,316 | 218,821 | 14.98% | 170 | 229,773 | 20.73% | 186 | 234,813 | 23.38% | 176 | 229,357 | 20.51% | 181 |
| X-n599-k92 | 598 | 108,451 | 119,969 | 10.62% | 96 | 135,041 | 24.52% | 104 | 126,374 | 16.53% | 112 | 126,954 | 17.06% | 104 |
| X-n613-k62 | 612 | 59,535 | 67,872 | 14.00% | 65 | 83,272 | 39.87% | 68 | 66,044 | 10.93% | 63 | 73,423 | 23.33% | 68 |
| X-n627-k43 | 626 | 62,164 | 84,115 | 35.31% | 60 | 72,166 | 16.09% | 46 | 68,418 | 10.06% | 44 | 76,172 | 22.53% | 46 |
| X-n641-k35 | 640 | 63,684 | 73,354 | 15.18% | 37 | 75,894 | 19.17% | 36 | 70,621 | 10.89% | 35 | 81,248 | 27.58% | 37 |
| X-n655-k131 | 654 | 106,780 | 114,546 | 7.27% | 131 | 117,545 | 10.08% | 132 | 140,279 | 31.37% | 157 | 134,342 | 25.81% | 138 |
| X-n670-k130 | 669 | 146,332 | 182,411 | 24.66% | 167 | 201,695 | 37.83% | 166 | 173,661 | 18.68% | 143 | 196,539 | 34.31% | 168 |
| X-n685-k75 | 684 | 68,205 | 77,963 | 14.31% | 77 | 93,000 | 36.35% | 85 | 74,822 | 9.70% | 77 | 84,631 | 24.08% | 82 |
| X-n701-k44 | 700 | 81,923 | 96,110 | 17.32% | 48 | 97,131 | 18.56% | 46 | 89,177 | 8.85% | 44 | 98,766 | 20.56% | 46 |
| X-n716-k35 | 715 | 43,373 | 52,639 | 21.36% | 36 | 53,664 | 23.73% | 36 | 48,871 | 12.68% | 35 | 55,771 | 28.58% | 36 |
| X-n733-k159 | 732 | 136,187 | 153,284 | 12.55% | 174 | 181,126 | 33.00% | 191 | 151,089 | 10.94% | 162 | 167,315 | 22.86% | 186 |
| X-n749-k98 | 748 | 77,269 | 89,932 | 16.39% | 106 | 101,012 | 30.73% | 113 | 85,709 | 10.92% | 99 | 96,446 | 24.82% | 110 |
| X-n766-k71 | 765 | 114,417 | 132,348 | 15.67% | 81 | 157,026 | 37.24% | 87 | 129,069 | 12.81% | 73 | 149,981 | 31.08% | 82 |
| X-n783-k48 | 782 | 72,386 | 86,078 | 18.92% | 50 | 89,099 | 23.09% | 52 | 81,152 | 12.11% | 49 | 91,564 | 26.49% | 52 |
| X-n801-k40 | 800 | 73,311 | 90,092 | 22.89% | 46 | 88,727 | 21.03% | 43 | 83,338 | 13.68% | 41 | 93,902 | 28.09% | 41 |
| X-n819-k171 | 818 | 158,121 | 175,808 | 11.19% | 180 | 202,528 | 28.08% | 199 | 216,694 | 37.04% | 185 | 189,575 | 19.89% | 193 |
| X-n837-k142 | 836 | 193,737 | 219,511 | 13.30% | 154 | 238,326 | 23.02% | 157 | 246,089 | 27.02% | 195 | 228,053 | 17.71% | 156 |
| X-n856-k95 | 855 | 88,965 | 100,889 | 13.40% | 97 | 115,565 | 29.90% | 101 | 98,373 | 10.57% | 96 | 113,843 | 27.96% | 100 |
| X-n876-k59 | 875 | 99,299 | 113,052 | 13.85% | 61 | 118,520 | 19.36% | 61 | 108,810 | 9.58% | 59 | 118,735 | 19.57% | 62 |
| X-n895-k37 | 894 | 53,860 | 65,832 | 22.23% | 40 | 69,602 | 29.23% | 41 | 63,755 | 18.37% | 41 | 78,711 | 46.14% | 42 |
| X-n916-k207 | 915 | 329,179 | 374,664 | 13.82% | 231 | 400,325 | 21.61% | 234 | 413,647 | 25.66% | 247 | 399,655 | 21.41% | 232 |
| X-n936-k151 | 935 | 132,715 | 175,354 | 32.13% | 189 | 195,322 | 47.17% | 200 | 159,656 | 20.30% | 172 | 193,911 | 46.11% | 204 |
| X-n957-k87 | 956 | 85,465 | 113,892 | 33.26% | 111 | 113,904 | 33.28% | 93 | 97,217 | 13.75% | 88 | 117,520 | 37.51% | 96 |
| X-n979-k58 | 978 | 118,976 | 161,230 | 35.51% | 145 | 136,956 | 15.11% | 61 | 130,532 | 9.71% | 60 | 141,094 | 18.59% | 61 |
| X-n1001-k43 | 1000 | 72,355 | 95,318 | 31.74% | 54 | 92,519 | 27.87% | 45 | 83,264 | 15.08% | 43 | 98,362 | 35.94% | 45 |

We evaluate the four models trained on 100-customer instances on all 100 CVRPLIB X-set instances, which contain between 100 and 1,000 customers. AM uses greedy decoding, whereas POMO uses one starting point per customer. Both methods use eightfold geometric augmentation. Objective values use the integer `EUC_2D` convention. The reported gap is computed against the reference cost in the solution file accompanying each instance. Every returned route set passes independent checks of customer coverage, capacity feasibility, and objective value.

Direct obtains a lower objective value on 90 of the 100 instances for AM and on 91 instances for POMO. On average, replacing direct route construction with customer ordering followed by Split increases the paired objective value by 5.09% for AM and 7.94% for POMO. The corresponding increases in the mean gap are 5.586 and 8.635 percentage points. Split also uses more vehicles on average for both backbones. The exceptions in the paired comparisons show that neither representation is uniformly superior on every instance, but they do not reverse the aggregate ordering.

**CVRPLIB XML100.** All four models return feasible solutions for all 10,000 XML100 instances and pass the independent verification of customer coverage, capacity feasibility, and objective value under the integer `EUC_2D` convention. As shown in Table 7, direct route construction achieves a lower mean cost and a lower mean optimality gap for both neural architectures. Compared with Split decoding, the mean gap is reduced by 3.294 percentage points for AM and by 4.263 percentage points for POMO. The corresponding reductions in mean cost are 2.71% and 3.74%, respectively.

The paired instance-level comparisons show that these differences are not driven by a small number of difficult instances. Direct AM obtains a lower objective value on 8,435 instances, whereas AM with Split decoding is better on 1,557 instances, with eight ties. Direct POMO is better on 9,045 instances, whereas POMO with Split decoding is better on 953 instances, with two ties. Direct route construction also uses fewer vehicles on average for both architectures.

These results extend the conclusion obtained on the synthetic CVRP test distribution to a substantially more heterogeneous benchmark. XML100 varies customer coordinates, depot locations, demand distributions, and expected route lengths. Nevertheless, generating a customer permutation and applying Split decoding does not improve average solution quality over direct route construction.

Table 5: Aggregate results on all 100 CVRPLIB X-set instances. The mean and median gaps are computed from the per-instance gaps. Lower values are better.

| Method | Feasible | Mean gap | Median gap | Mean vehicles |
|---|---|---|---|---|
| AM Direct | 100/100 | 13.540% | 11.623% | 56.20 |
| AM Split | 100/100 | 19.126% | 17.166% | 56.72 |
| POMO Direct | 100/100 | 10.370% | 8.940% | 54.23 |
| POMO Split | 100/100 | 19.005% | 17.945% | 56.18 |

Table 7: Results on all 10,000 CVRPLIB XML100 instances. Objective values are computed using the integer `EUC_2D` convention. The gap is computed against the proven optimum of each instance before averaging. AM uses greedy decoding, whereas POMO uses 100 starting points. Both methods use eightfold geometric augmentation. All 40,000 model–instance solutions are feasible. Lower cost and gap are better.

| Method | Feasible | Vehicles | Mean cost | Mean gap |
|---|---|---|---|---|
| AM Direct | 10,000/10,000 | 12.707 | 18,391.55 | 8.727% |
| AM Split | 10,000/10,000 | 12.911 | 18,890.54 | 12.022% |
| POMO Direct | 10,000/10,000 | 12.422 | 17,926.07 | 5.904% |
| POMO Split | 10,000/10,000 | 12.711 | 18,597.31 | 10.167% |

**Solomon CVRPTW benchmark.** All 448 model–instance solutions are feasible and pass the independent verification of customer coverage, duplicate visits, vehicle capacity, customer time windows, return to the depot, and total travel distance. Table 8 shows that direct route construction achieves a lower mean distance in three of the four comparisons. At 50 customers, Split decoding increases the mean distance by 9.80% for AM and by 4.71% for POMO. At 100 customers, AM with Split decoding remains 5.92% worse than its direct counterpart.

POMO at 100 customers is the only aggregate reversal. In this setting, Split decoding reduces the mean distance from 1,646.48 to 1,451.98, corresponding to an improvement of 11.81%. This reduction is accompanied by an increase in the mean number of vehicles from 13.48 to 14.75. The improvement is also not uniform across the six Solomon instance classes. Split decoding obtains a lower mean distance on C1, C2, R2, and RC2, whereas direct construction obtains a lower mean distance on R1 and RC1. At the instance level, the Split variant is better on 35 of the 56 instances, while the direct variant is better on the remaining 21.

The other three comparisons favor direct construction on most individual instances. Direct AM is better on 39 of the 56 instances at 50 customers and on 32 of the 56 instances at 100 customers. Direct POMO is better on 40 of the 56 instances at 50 customers. Across all four paired comparisons, direct construction obtains a lower distance on 132 of the 224 paired instances, whereas Split decoding obtains a lower distance on 92. The Split variants also use more vehicles on average in every comparison.

Candidate solutions in this experiment are selected solely according to total travel distance. The number of vehicles is reported as an additional measure but is not minimized before distance. The results therefore should not be interpreted according to the standard lexicographic evaluation that first minimizes the number of vehicles and then minimizes total distance. Instead, they evaluate the quality of the learned customer order and the resulting route partition under a distance-based objective.

Table 8: Average results over all 56 Solomon instances. AM uses greedy decoding, whereas POMO uses one starting point per customer. Both methods use eightfold geometric augmentation, and candidate solutions are selected according to total travel distance. The number of vehicles is reported separately and is not used for candidate selection. No upper bound on the number of vehicles is imposed. All 448 model–instance solutions are feasible and pass independent route verification. Lower distance is better.

| | | Direct | | Split | |
|---|---|---|---|---|---|
| Backbone | $n$ | Distance | Vehicles | Distance | Vehicles |
| AM | 50 | 844.89 | 6.39 | 927.67 | 7.82 |
| AM | 100 | 1,695.87 | 9.96 | 1,796.25 | 15.30 |
| POMO | 50 | 753.97 | 6.63 | 789.44 | 7.79 |
| POMO | 100 | 1,646.48 | 13.48 | 1,451.98 | 14.75 |

Table 9: Solomon-50 results by instance class for the latest AM and POMO same-size models under distance-first candidate selection. A Direct cell reports mean vehicles/mean distance. A Split cell also reports its relative distance difference from the matched Direct model in parentheses.

| Method | C1 | C2 | R1 | R2 | RC1 | RC2 |
|---|---|---|---|---|---|---|
| AM Direct | 7.89/689.3 | 2.62/722.6 | 9.25/872.5 | 5.18/1002.5 | 7.50/871.1 | 4.75/858.0 |
| AM Split | 8.11/758.4 (+10.03%) | 6.62/1103.8 (+52.75%) | 10.75/899.7 (+3.12%) | 5.82/995.8 (-0.67%) | 9.12/942.9 (+8.25%) | 5.75/875.0 (+1.98%) |
| POMO Direct | 7.56/548.3 | 5.50/749.0 | 8.92/823.6 | 5.09/847.5 | 7.50/830.7 | 4.50/680.5 |
| POMO Split | 7.44/527.6 (-3.78%) | 7.50/820.1 (+9.50%) | 10.17/857.5 (+4.12%) | 6.64/835.5 (-1.42%) | 8.50/881.6 (+6.13%) | 5.75/795.8 (+16.94%) |

Table 10: Per-instance Solomon-50 results under distance-first candidate selection. Direct cells report vehicles/distance. Split cells also report the relative distance difference from the matched Direct model. The reference prioritizes vehicle count before distance and is included for context only.

| Instance | Class | Reference | AM Direct | AM Split | POMO Direct | POMO Split |
|---|---|---|---|---|---|---|
| C101 | C1 | 5/362.40 | 7/551.30 | 10/855.56 (+55.19%) | 8/497.86 | 8/504.27 (+1.29%) |
| C102 | C1 | 5/361.40 | 9/781.58 | 8/833.81 (+6.68%) | 8/583.28 | 8/583.87 (+0.10%) |
| C103 | C1 | 5/361.40 | 7/770.53 | 7/777.71 (+0.93%) | 6/561.70 | 8/544.75 (-3.02%) |
| C104 | C1 | 5/358.00 | 5/664.63 | 8/685.64 (+3.16%) | 6/595.72 | 7/546.80 (-8.21%) |
| C105 | C1 | 5/362.40 | 9/669.87 | 9/840.58 (+25.48%) | 8/498.41 | 8/535.84 (+7.51%) |
| C106 | C1 | 5/362.40 | 7/691.85 | 9/900.49 (+30.16%) | 7/438.91 | 8/553.08 (+26.01%) |
| C107 | C1 | 5/362.40 | 10/621.45 | 7/683.81 (+10.03%) | 7/569.45 | 7/525.09 (-7.79%) |
| C108 | C1 | 5/362.40 | 8/795.93 | 8/630.58 (-20.77%) | 8/567.63 | 7/534.37 (-5.86%) |
| C109 | C1 | 5/362.40 | 9/656.33 | 7/617.69 (-5.89%) | 10/622.17 | 6/420.54 (-32.41%) |
| C201 | C2 | 3/360.20 | 2/484.46 | 9/1236.35 (+155.20%) | 7/782.98 | 10/934.33 (+19.33%) |
| C202 | C2 | 3/360.20 | 2/672.41 | 7/1150.68 (+71.13%) | 5/820.02 | 8/893.49 (+8.96%) |
| C203 | C2 | 3/359.80 | 3/878.26 | 6/1111.43 (+26.55%) | 3/888.49 | 8/784.50 (-11.70%) |
| C204 | C2 | 2/350.10 | 3/809.11 | 4/918.62 (+13.53%) | 8/874.40 | 4/637.35 (-27.11%) |
| C205 | C2 | 3/359.80 | 2/535.12 | 8/1174.91 (+119.56%) | 5/565.34 | 8/915.78 (+61.99%) |
| C206 | C2 | 3/359.80 | 3/837.19 | 6/1061.08 (+26.74%) | 5/696.38 | 8/895.46 (+28.59%) |
| C207 | C2 | 3/359.60 | 3/821.98 | 6/1101.10 (+33.96%) | 4/683.51 | 7/793.65 (+16.11%) |
| C208 | C2 | 2/350.50 | 3/742.53 | 7/1076.59 (+44.99%) | 7/680.64 | 7/706.17 (+3.75%) |
| R101 | R1 | 12/1044.00 | 13/1107.63 | 17/1204.77 (+8.77%) | 13/1052.39 | 16/1136.57 (+8.00%) |
| R102 | R1 | 11/909.00 | 11/986.96 | 13/1038.14 (+5.19%) | 11/953.78 | 12/949.59 (-0.44%) |
| R103 | R1 | 9/772.90 | 10/869.31 | 11/932.98 (+7.32%) | 9/846.75 | 11/889.13 (+5.00%) |
| R104 | R1 | 6/625.40 | 8/781.40 | 8/739.51 (-5.36%) | 7/713.01 | 8/746.11 (+4.64%) |
| R105 | R1 | 9/899.30 | 11/991.27 | 14/1040.82 (+5.00%) | 11/967.23 | 11/973.09 (+0.61%) |
| R106 | R1 | 8/793.00 | 10/926.86 | 12/952.18 (+2.73%) | 9/883.99 | 11/878.56 (-0.61%) |
| R107 | R1 | 7/711.10 | 8/825.77 | 10/865.93 (+4.86%) | 8/780.48 | 9/825.65 (+5.79%) |

Continued on next page

Table 10 continued

| Instance | Class | Reference | AM Direct | AM Split | POMO Direct | POMO Split |
|---|---|---|---|---|---|---|
| R108 | R1 | 6/617.70 | 7/762.97 | 8/745.82 (-2.25%) | 7/702.95 | 7/707.12 (+0.59%) |
| R109 | R1 | 8/786.80 | 9/888.65 | 10/913.26 (+2.77%) | 9/828.12 | 10/887.65 (+7.19%) |
| R110 | R1 | 7/697.00 | 9/785.42 | 10/838.07 (+6.70%) | 8/730.82 | 10/800.73 (+9.57%) |
| R111 | R1 | 7/707.20 | 8/812.27 | 8/800.69 (-1.43%) | 8/755.18 | 9/791.37 (+4.79%) |
| R112 | R1 | 6/630.20 | 7/731.12 | 8/723.71 (-1.01%) | 7/668.85 | 8/704.86 (+5.38%) |
| R201 | R2 | 6/791.90 | 6/1105.70 | 10/1223.50 (+10.65%) | 7/1036.70 | 10/1012.64 (-2.32%) |
| R202 | R2 | 5/698.50 | 8/1166.28 | 7/1128.91 (-3.20%) | 6/964.96 | 8/948.38 (-1.72%) |
| R203 | R2 | 5/605.30 | 5/1041.43 | 5/973.12 (-6.56%) | 4/874.07 | 7/794.63 (-9.09%) |
| R204 | R2 | 2/506.40 | 3/815.41 | 5/836.44 (+2.58%) | 3/688.38 | 4/689.32 (+0.14%) |
| R205 | R2 | 4/690.10 | 7/1052.99 | 6/1158.00 (+9.97%) | 6/881.71 | 8/921.02 (+4.46%) |
| R206 | R2 | 4/632.40 | 6/1044.71 | 5/910.47 (-12.85%) | 6/861.67 | 7/874.02 (+1.43%) |
| R207 | R2 | 3/575.50 | 4/998.66 | 5/964.99 (-3.37%) | 4/869.27 | 6/776.37 (-10.69%) |
| R208 | R2 | 2/487.70 | 3/829.29 | 4/779.84 (-5.96%) | 3/694.08 | 3/692.34 (-0.25%) |
| R209 | R2 | 4/600.60 | 5/1020.48 | 6/1017.58 (-0.28%) | 7/832.57 | 8/861.12 (+3.43%) |
| R210 | R2 | 4/645.60 | 6/1065.96 | 6/1045.89 (-1.88%) | 5/862.50 | 7/872.15 (+1.12%) |
| R211 | R2 | 3/535.50 | 4/886.54 | 5/914.83 (+3.19%) | 5/756.60 | 5/748.29 (-1.10%) |
| RC101 | RC1 | 8/944.00 | 9/1011.95 | 12/1164.84 (+15.11%) | 10/1021.81 | 10/986.77 (-3.43%) |
| RC102 | RC1 | 7/822.50 | 8/948.17 | 9/958.34 (+1.07%) | 8/909.52 | 9/931.82 (+2.45%) |
| RC103 | RC1 | 6/710.90 | 8/895.90 | 9/938.52 (+4.76%) | 7/844.13 | 9/917.73 (+8.72%) |
| RC104 | RC1 | 5/545.80 | 6/690.44 | 8/770.43 (+11.58%) | 6/629.97 | 7/708.76 (+12.51%) |
| RC105 | RC1 | 8/855.30 | 9/978.98 | 11/1072.25 (+9.53%) | 8/911.07 | 9/933.59 (+2.47%) |
| RC106 | RC1 | 6/723.20 | 7/863.02 | 9/968.84 (+12.26%) | 8/872.73 | 9/932.17 (+6.81%) |
| RC107 | RC1 | 6/642.70 | 6/801.48 | 8/888.15 (+10.81%) | 7/746.94 | 8/849.71 (+13.76%) |
| RC108 | RC1 | 6/598.10 | 7/778.65 | 7/782.12 (+0.45%) | 6/709.11 | 7/792.17 (+11.71%) |
| RC201 | RC2 | 5/684.80 | 5/918.78 | 10/1177.55 (+28.17%) | 5/745.67 | 8/958.84 (+28.59%) |
| RC202 | RC2 | 5/613.60 | 7/1026.00 | 5/879.79 (-14.25%) | 5/716.70 | 6/857.46 (+19.64%) |
| RC203 | RC2 | 4/555.30 | 5/818.13 | 5/772.11 (-5.63%) | 4/693.53 | 6/758.65 (+9.39%) |
| RC204 | RC2 | 3/444.20 | 3/737.89 | 4/697.38 (-5.49%) | 3/539.37 | 3/562.02 (+4.20%) |
| RC205 | RC2 | 5/630.20 | 6/984.64 | 7/1007.40 (+2.31%) | 5/723.64 | 7/943.24 (+30.35%) |
| RC206 | RC2 | 5/610.00 | 4/852.94 | 6/963.09 (+12.91%) | 5/686.98 | 6/871.38 (+26.84%) |
| RC207 | RC2 | 4/558.60 | 4/808.40 | 6/884.82 (+9.45%) | 5/707.68 | 6/774.01 (+9.37%) |
| RC208 | RC2 | 3/476.70 | 4/716.93 | 3/617.72 (-13.84%) | 4/630.28 | 4/640.55 (+1.63%) |

**Representative route visualizations.** Figures 5 and 6 illustrate how the customer order affects the solution obtained after Split decoding. These examples are presented for qualitative interpretation; the quantitative conclusions are based on the complete benchmark results reported in the corresponding tables.

On C205, the Split solution contains more routes and several long connections between spatially separated customers. Although Split finds the minimum-cost feasible contiguous partition for the predicted customer order, it cannot change the relative order of customers. The increase in both distance and vehicle count therefore reflects the limited quality of the order supplied to Split rather than a failure of the partitioning algorithm or a violation of the constraints. This example also shows why feasibility alone is insufficient for evaluating a customer-ordering policy: an order may admit a feasible partition while still inducing inefficient routes.

The XML100 example isolates the effect of customer ordering more clearly because Direct and Split use the same number of vehicles. The Split solution contains longer connections and less compact tours within the three routes, while the Direct policy produces a customer order that is better aligned with the spatial structure of the instance. Since Split can optimize only the route boundaries of a fixed order, it cannot remove these detours by reordering customers. Together, the two examples illustrate two distinct sources of additional cost: an unsuitable order may increase the number of routes, as on C205, or reduce route compactness even when the vehicle count is unchanged, as on XML100_1216_12.

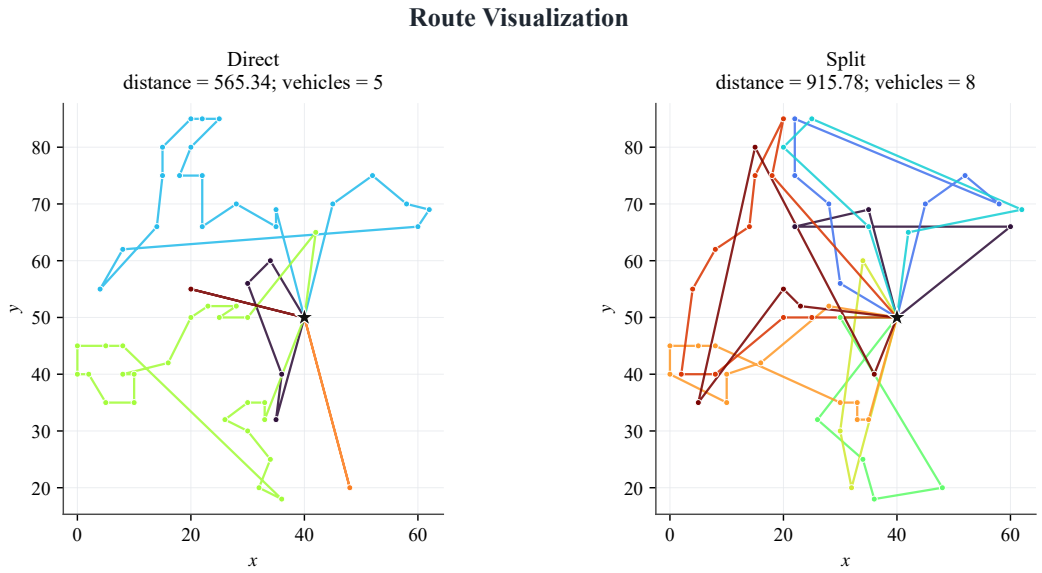



Figure 5: Route visualization for POMO on the Solomon-50 instance C205. Both solutions are feasible and are selected by total distance without first minimizing the number of vehicles. Direct construction obtains a distance of 565.34 using five vehicles, whereas the customer order produced by the Split variant yields a distance of 915.78 using eight vehicles, corresponding to a 61.99% increase. Different colors denote different vehicle routes, and the star denotes the depot.

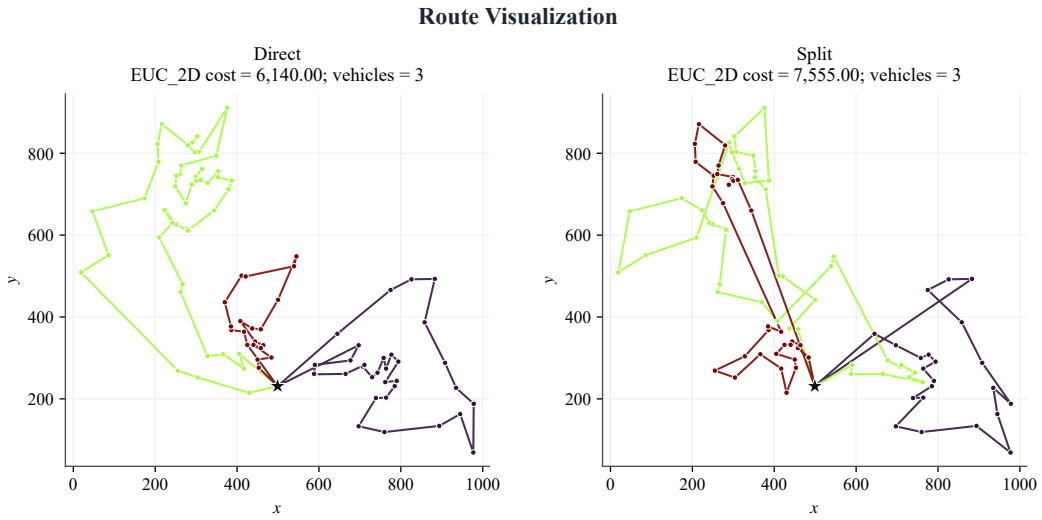



Figure 6: Route visualization for POMO on the CVRPLIB XML100 instance XML100_1216_12. Costs follow the official integer EUC_2D convention. Direct construction obtains a cost of 6,140, whereas the Split variant obtains 7,555, corresponding to a 23.05% increase. Both solutions use three vehicles, so the difference cannot be attributed to the number of routes. Different colors denote different vehicle routes, and the star denotes the depot.

## E LIMITATIONS

The conclusions are limited to the neural construction methods, problem variants, and inference procedures evaluated in this study. The customer-ordering policies retain the same static node features and the same types of dynamic decoder variables as their direct route construction counterparts. However, because they do not select the depot, the dynamic variables evolve continuously along

the customer permutation and are not reset at route boundaries during permutation generation. The capacity and time-window feasibility masks used in the single-task direct policies are not applied. In the multi-task setting, the policy retains the original decoder variables and uses an additional action mask for backhaul feasibility, whereas capacity, hard time windows, open routes, and route duration limits are enforced only by Split. The results therefore do not rule out other customer-ordering policies with different state transitions, feasibility masks, or mechanisms for revising the predicted order.

Split returns a minimum-cost feasible contiguous partition only when such a partition exists for the fixed customer order. It cannot change the relative order of customers and should not be interpreted as a replacement for improvement methods that modify the order. Under some combinations of backhauls and hard time windows, the generated customer order does not admit a feasible contiguous partition even though the backhaul action mask is applied during decoding. In addition, the general Split implementation has quadratic time complexity, which may become a computational limitation at substantially larger problem sizes.

The empirical evidence has several further limitations. We report one completed training run for each model and therefore do not estimate variation across random seeds. In the multi-task study, the direct route construction references are published aggregate results rather than paired outputs on the same saved instances, which prevents instance-level statistical comparison. The variants combining backhauls with hard time windows are excluded from the solution-quality tables because most generated customer orders do not admit a feasible contiguous partition. Their low feasibility rates are reported as a limitation of generalization to unseen combinations of constraints rather than as objective values averaged conditionally over the small subset of feasible solutions. Finally, the Solomon experiments select candidate solutions according to total travel distance and report the number of vehicles separately. They should therefore not be compared directly with benchmark results that first minimize the number of vehicles and then minimize distance. These limitations restrict the scope of the empirical conclusions, but they do not affect the independently verified feasibility and objective values of the complete solutions reported in the corresponding tables.