GENROUTE: A MINIMAL CONSTRUCTIVE NEURAL SOLVER FOR MULTI-DEPOT VEHICLE ROUTING VARIANTS VIA DEPOT-CONDITIONED DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

The multi-depot vehicle routing problem asks how vehicles operating from several depots should serve customers at minimum travel cost. Neural methods can construct solutions quickly, but existing approaches handle multiple depots and operational constraints by adding specialized encoders, expert networks, or fine-tuning a separate model for each variant. We show that a substantially simpler design is sufficient. Building on the Attention-Model/POMO line of constructive solvers, our first contribution is that multi-depot structure needs no specialized machinery: it is enough for the decoder's standard context to carry the currently active depot. This induces an implicit sequential decomposition: a shared encoder represents the complete instance, while the decoder constructs one depot-centered part of the solution at a time, jointly deciding customer allocation and routing without explicit clustering. Each encoder layer augments its attention logits with a lightweight learned bias computed from pairwise distances. A three-stage curriculum first trains the core multi-depot problem and then progressively expands the same architecture into a single model covering 16 combinations of open routes, mixed backhauls, route-length limits, and time windows. On held-out evaluations the core model attains a 2.26% gap to reference solutions, while the final generalist reaches a 3.95% mean gap across 16 multi-depot variants and the lowest gap in 9 of 12 depot-count/problem-size settings.

1 INTRODUCTION

The vehicle routing problem (VRP) and its multi-depot variant (MDVRP) are central to logistics and distribution planning. Because these problems are NP-hard, exact optimization becomes increasingly expensive as instance size grows. Hybrid genetic search methods provide high-quality practical solutions, with PyVRP offering an efficient open-source implementation for several VRP variants (Vidal et al., 2012; Vidal, 2022; Wouda et al., 2024). Such methods nevertheless require algorithm design and tuning for the problem under consideration. Neural combinatorial optimization offers a complementary learned-construction paradigm: Pointer Networks and early reinforcement-learning policies established learned routing construction, followed by the Attention Model and POMO (Vinyals et al., 2015; Bello et al., 2017; Nazari et al., 2018; Kool et al., 2019; Kwon et al., 2020). These works support fast learned heuristic construction, but do not imply that neural solvers generally replace classical heuristics.

Multi-depot and cross-problem neural solvers introduce different mechanisms to represent depot–customer structure and operational constraints. Examples include multi-type embeddings and attention in MD-MTA (Li et al., 2024), edge-aware attention with transfer learning and local search in TuneNSearch (Corrêa et al., 2026b), mixture-of-experts in MVMoE (Zhou et al., 2024), adapters in RouteFinder (Berto et al., 2025), basis-variant LoRA experts in MoSES (Pan et al., 2025), and constraint-conditioned feature modulation in FiLMMeD (Corrêa et al., 2026a). Each of these mechanisms is specific to depots or to variants, whether as separate embedding streams, an explicit assignment stage(Nasehi et al., 2024), experts, adapters, modulation layers, or target-specific fine-tuning with local search. GenRoute uses none of them: the depot that owns the route under construction enters the decoder context as its encoded token, and the rest of the model is the shared encoder and

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

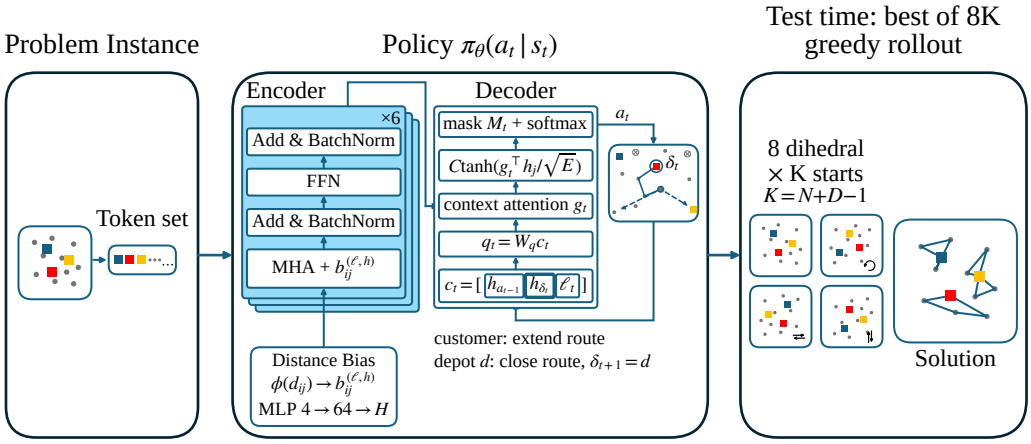


Figure 1: GenRoute's single-stream construction. All depots, customers, and attributes enter one shared encoder; the learned distance term is an additive bias on every encoder attention logit. The central mechanism is the decoder context: the last node, the *active depot* δ_t, and the remaining load condition every decision. Selecting a customer appends it to the current depot-owned route; selecting depot d closes that route and sets $\delta_{t+1} = d$. One policy therefore performs customer allocation and routing jointly, with feasibility enforced by the mask.

pointer decoder of the Attention Model and POMO(Kool et al., 2019; Kwon et al., 2020), with a learned distance bias on the encoder attention.

This paper asks whether multi-depot structure requires such specialized machinery. Fig. 1 summarizes our answer: depots, customers, and their attributes form one shared token set; a pairwise-distance-biased encoder produces their representations; and the decoder combines the last node, the active-depot representation, and the remaining load to choose the next feasible action. Customer actions extend the current depot-owned route, whereas depot actions close it and start the next route, so one action sequence jointly allocates customers and constructs routes; at test time, the shortest solution across geometric transforms and POMO starts is retained. We train this single-stream constructor from scratch with REINFORCE (Williams, 1992), without clustering, transfer learning, or local search, using the Attention Model and POMO as its backbone (Kool et al., 2019; Kwon et al., 2020). Route-origin information itself is not new: RouteFinder uses origin-depot coordinates and DeepMDV uses a depot embedding (Berto et al., 2025; Nasehi et al., 2024). Our claim is that an encoded active-depot token, combined with depot actions as route delimiters, enables joint allocation and routing in one shared constructor.

The contributions of this work are threefold. (1) A minimal single-stream multi-depot constructor that uses explicit active-depot conditioning to jointly allocate customers and construct routes. (2) The integration of a Graphormer-style learned pairwise-distance bias(Ying et al., 2021) into every shared encoder layer of a multi-depot constructor, adding $8{,}160$ parameters and no edge embeddings, which improves generalization across problem sizes and instance layouts. (3) A three-stage training procedure that expands the same architecture, with every variant input present from the first stage, into a single generalist model covering 16 combinations of open routes, mixed backhauls, route-length limits, and time windows and depot counts $D \in \{1, 2, 3\}$.

2 Related Work

Constructive neural routing. The Attention Model and POMO provide the encoder–decoder and multi-start backbone used here, while Sym-NCO further exploits routing symmetries (Kool et al., 2019; Kwon et al., 2020; Kim et al., 2022). Pairwise geometry has been modeled through matrix-valued attention in MatNet and distance-based attention modification in DAR (Kwon et al., 2021; Wang et al., 2024); neural routing policies nevertheless remain sensitive to distribution shift (Joshi et al., 2022).

Multi-depot solvers. DeepMDV learns customer-to-tour assignments before routing each tour separately and includes the starting-depot embedding in its active-tour state (Nasehi et al., 2024). MD-MTA instead constructs solutions jointly using depot, customer, and all-node embedding streams with multi-type attention (Li et al., 2024), while TuneNSearch combines an edge-aware encoder with transfer learning and local search (Corrêa et al., 2026b). GenRoute studies whether multi-depot structure can be captured by an otherwise standard one-stage constructor.

Cross-problem and variable-depot training. MVMoE, RouteFinder, MoSES, FiLMMeD, and CaDA respectively explore expert routing, attribute/adaptor mechanisms, specialized low-rank experts, feature modulation, and constraint-aware attention for multi-variant solving (Zhou et al., 2024; Berto et al., 2025; Pan et al., 2025; Corrêa et al., 2026a; Li et al., 2025). URS uses a unified data representation and evaluates multi-depot variants zero-shot rather than training on them (Zhou et al., 2026). RouteFinder supplies raw origin-depot coordinates to its decoder, whereas DeepMDV uses an origin-depot embedding; GenRoute places the encoded active-depot token in a minimal shared decoder and evaluates it against a controlled depot-blind Stage-1 reference. Mixed depot-count training is also not unprecedented: DPN samples $D \in \{2, \dots, 10\}$ for min–max routing (Zheng et al., 2024). Our mixed-D contribution is one min–sum checkpoint trained over $D \in \{1, 2, 3\}$ and all 16 operational variants.

3 Problem Formulation

We consider the relaxed Euclidean multi-depot capacitated vehicle routing problem used by the neural baselines in our experiments. An instance is a tuple $x = (V, p, q, Q)$ with node set $V = V_{\text{dep}} \cup V_{\text{cus}}$, where $V_{\text{dep}} = \{1, \dots, D\}$ contains the depots and $V_{\text{cus}} = \{D{+}1, \dots, D{+}N\}$ contains the customers. Each node i has coordinates $p_i \in [0, 1]^2$, each customer has demand $q_i > 0$, and Q is the vehicle capacity. Let $d_{ij} = \|p_i - p_j\|_2$. A solution is a set of routes $\mathcal{R} = \{r_1, \dots, r_m\}$, where each route $r = (\delta, c_1, \dots, c_{|r|}, \delta)$ starts and ends at the same depot $\delta \in V_{\text{dep}}$. Every customer must appear in exactly one route, and each route must satisfy $\sum_{c \in r} q_c \le Q$. The objective is to minimize the total Euclidean route length, $L(\mathcal{R}) = \sum_{r \in \mathcal{R}} L(r)$. We impose no fixed limit on the number of routes or vehicles per depot. This is a relaxation of classical MDVRP formulations that may impose fixed fleets and instance-specific route-duration limits (Cordeau et al., 1997; Vidal et al., 2012).

The generalist covers all 16 combinations of four attributes. Open routes (O) omit the return edge from the final customer to the route's depot. In mixed-backhaul variants (MB, denoted B in our tables), customers with $q_i < 0$ represent pickups; deliveries and pickups may be interleaved without precedence, while their carried loads are tracked separately and bounded by Q. A route-length limit (L) bounds the total travel length of each route, including the return edge for closed routes. With time windows (TW), each node has a hard window $[e_i, l_i]$ and service time s_i; early arrival incurs waiting, service must begin within the window, and a closed route must return to its depot before the horizon. The CVRP is recovered when $D = 1$. TSP is not part of the 16-variant training family.

4 Methodology

Construction MDP. Given an instance x, the policy π_θ constructs a solution as an action sequence $\tau = (a_1, \dots, a_T)$. At step t, the base MDCVRP state is $s_t = (a_{t-1}, \delta_t, \ell_t, U_t)$, where a_{t-1} is the last selected node, δ_t is the *active depot* that owns the route under construction, $\ell_t \in [0, Q]$ is the remaining vehicle capacity, and U_t is the set of unvisited customers. The action space contains all $D + N$ nodes. Selecting a feasible customer j appends it to the current route and updates $\ell_{t+1} = \ell_t - q_j$ and $U_{t+1} = U_t \setminus \{j\}$. Selecting a depot d closes the current route by adding the return edge from the last customer to its owning depot δ_t, and then starts the next route at d, with $\delta_{t+1} = d$ and $\ell_{t+1} = Q$. No travel cost from δ_t to d is added: a depot action is a route delimiter and next-route selector, not a physical relocation between depots. Visited customers, customers with $q_j > \ell_t$, and consecutive depot actions are masked. The episode ends after every customer has been visited; the final open route is then closed automatically, except in open-route variants where its return edge is omitted. The reward is $R(\tau) = -L(\tau)$.

4.1 Model Architecture

Figure 1 gives an overview of the network.

Input embeddings. For the base MDCVRP model, depots and customers are projected into a shared E-dimensional space ($E = 256$) using two linear maps:

$$h_i^{(0)} = \begin{cases} W^{\text{dep}} p_i + b^{\text{dep}}, & i \in V_{\text{dep}}, \\ W^{\text{cus}}[\, p_i;\, q_i \,] + b^{\text{cus}}, & i \in V_{\text{cus}}. \end{cases} \tag{1}$$

For the final generalist, the customer input is extended to $[\, p_i;\, |q_i|;\, e_i;\, l_i;\, s_i;\, b_i;\, o \,]$, where b_i indicates a backhaul customer and $o \in \{0, 1\}$ indicates an open-route instance. Attributes that are inactive for a variant are set to zero. All $D + N$ depot and customer tokens are then processed together by one shared encoder stack; there are no depot-specific or customer-specific attention streams.

Distance-biased attention. The encoder contains $L_{\text{enc}} = 6$ POMO-style layers with $H = 16$ attention heads and key dimension $d_k = 16$. Each encoder layer adds a learned geometric bias to its attention logits. The bias is inspired by Graphormer's spatial attention bias (Ying et al., 2021). Distance-aware attention reshaping also uses Euclidean distance for cross-scale routing generalization, but modifies decoder attention scores without additional training (Wang et al., 2024). In contrast, our encoder learns a separate per-head offset from every pairwise distance.

Writing d_{ij} for the Euclidean distance between nodes i and j, we compute once per instance

$$\phi(d_{ij}) = \left[d_{ij},\ d_{ij}^2,\ \log(d_{ij} + \varepsilon),\ (d_{ij} + \varepsilon)^{-1} \right], \tag{2}$$

where $\varepsilon = 10^{-8}$. Self-pairs use $d_{ii} = 0$ without masking, so their reciprocal feature equals $1/\varepsilon$; the zero-initialised output layer below leaves the network to learn how to weight it. Each encoder layer ℓ has a two-layer MLP with dimensions $4 \to 64 \to H$. Its output for head h is

$$b_{ij}^{(\ell,h)} = \left[\text{MLP}_\ell(\phi(d_{ij})) \right]_h . \tag{3}$$

Let $q_i^{(\ell,h)} = W_Q^{(\ell,h)} h_i^{(\ell-1)}$ and $k_j^{(\ell,h)} = W_K^{(\ell,h)} h_j^{(\ell-1)}$. The resulting scalar is added to the corresponding attention logit:

$$A_{ij}^{(\ell,h)} = \text{softmax}_j \left(\frac{\left(q_i^{(\ell,h)} \right)^\top k_j^{(\ell,h)}}{\sqrt{d_k}} + b_{ij}^{(\ell,h)} \right). \tag{4}$$

Each layer otherwise follows the POMO convention: multi-head attention with residual connection and batch normalization (statistics pooled over the batch and node dimensions), then a feed-forward block ($E \to 1024 \to E$, ReLU) with a second residual and normalization. When explicit pairwise geometry enters, it does so only through the scalar bias in Eq. (4), rather than through extra encoder streams (MD-MTA) or edge-embedding attention (TuneNSearch).

Zero-initialization and what is learned. The output layer of every distance MLP is initialized to zero. Consequently, the learned distance term initially contributes zero to every attention logit and learns during training how much geometric information to introduce. Across six encoder layers, the distance modules add $8{,}160$ parameters. At inference, the trained MLPs are applied directly to the distance matrix of each new instance, without parameter adaptation or additional optimization.

Decoder. Let $h_i = h_i^{(L_{\text{enc}})}$ denote the final encoder embedding of node i. The encoder runs once, after which a stateless pointer decoder is queried at every construction step. For the base MDCVRP model, its context is

$$c_t^{\text{base}} = \left[h_{a_{t-1}};\, h_{\delta_t};\, \ell_t \right] \in \mathbb{R}^{2E+1}, \tag{5}$$

where $h_{a_{t-1}}$ represents the last selected node, h_{δ_t} represents the active depot that owns the current route, and ℓ_t is the remaining vehicle capacity.

For the final generalist, five additional state values are appended:

$$c_t^{\text{gen}} = \left[h_{a_{t-1}};\, h_{\delta_t};\, \ell_t;\, \xi_t;\, \rho_t;\, \bar{\rho};\, \beta_t;\, o \right], \tag{6}$$

where ξ_t is elapsed route time, ρ_t is the current route length, $\bar{\rho}$ is the route-length limit, β_t is the collected backhaul load, and o is the open-route flag. The accumulated route length ρ_t is always provided as a general state variable. The elapsed time ξ_t, route-length limit $\bar{\rho}$, collected backhaul load β_t, and open-route flag o are zero when their corresponding attributes are inactive.

The active-depot embedding is the central mechanism studied in this work. The depot-blind reference replaces h_{δ_t} in the context by a zero vector while retaining the same environment transitions and feasibility masks. A multi-head glimpse projects the context as a query and attends over the cached keys and values of all $D + N$ encoded nodes, using the feasibility mask, to produce $g_t \in \mathbb{R}^E$. A single-head pointer then assigns the clipped logits

$$u_{t,j} = C \tanh\left(\frac{g_t^\top h_j}{\sqrt{E}}\right), \tag{7}$$

followed by

$$\pi_\theta(a_t = j \mid s_t) = \mathrm{softmax}_j(u_{t,j} + M_{t,j}), \tag{8}$$

where $C = 10$ and $M_{t,j} \in \{0, -\infty\}$ is the feasibility mask. The probability of a complete construction is $p_\theta(\tau \mid x) = \prod_{t=1}^{T} \pi_\theta(a_t \mid s_t)$. The complete model contains approximately 5.1 million parameters.

4.2 Training Objective

All three curriculum stages use REINFORCE with a POMO shared baseline (Williams, 1992; Kwon et al., 2020). For each unaugmented training instance, we sample the complete set of $K = N+D-1$ trajectories with distinct forced starts: N customer starts and the additional depot starts $2, \ldots, D$. Customer starts depart from depot 1, which is then the active depot; the forced first action updates the state but is excluded from the log-likelihood in Eq. (10). Their within-instance mean reward is

$$b(x) = \frac{1}{K} \sum_{k=1}^{K} R\left(\tau^{(k)}\right). \tag{9}$$

The resulting gradient estimator is

$$\nabla_\theta J \approx \frac{1}{K} \sum_{k=1}^{K} \left(R\left(\tau^{(k)}\right) - b(x)\right) \nabla_\theta \log p_\theta\left(\tau^{(k)} \mid x\right). \tag{10}$$

No learned critic is required. The distance-bias modules and all remaining parameters are optimized jointly from Stage 1 and retained throughout the curriculum. Batch sizes, learning rates, and stage schedules are given in the curriculum description below.

4.3 Inference

Inference is deterministic: at every step, the decoder selects $a_t = \arg\max_j \pi_\theta(j \mid s_t)$. For each transformed copy of an instance, we decode all $K = N + D - 1$ POMO starts and retain the solution with the smallest reconstructed route length. Using A geometric transforms, the selected solution is

$$\hat{\tau} = \underset{a \in \{1,\ldots,A\},\, k \in \{1,\ldots,K\}}{\arg\min} L\left(\tau^{(a,k)}\right). \tag{11}$$

Unless stated otherwise, we use the eight dihedral transformations ($A = 8$), giving $8K$ greedy rollouts and one encoder evaluation for each transformed copy. When augmentation is disabled in a paired comparison, we use $A = 1$ for every compared model and state this explicitly with the result.

The final generalist uses batch normalization without stored running statistics. We therefore evaluate it one instance at a time so that the solution for an instance cannot depend on the other instances placed in the same evaluation batch.

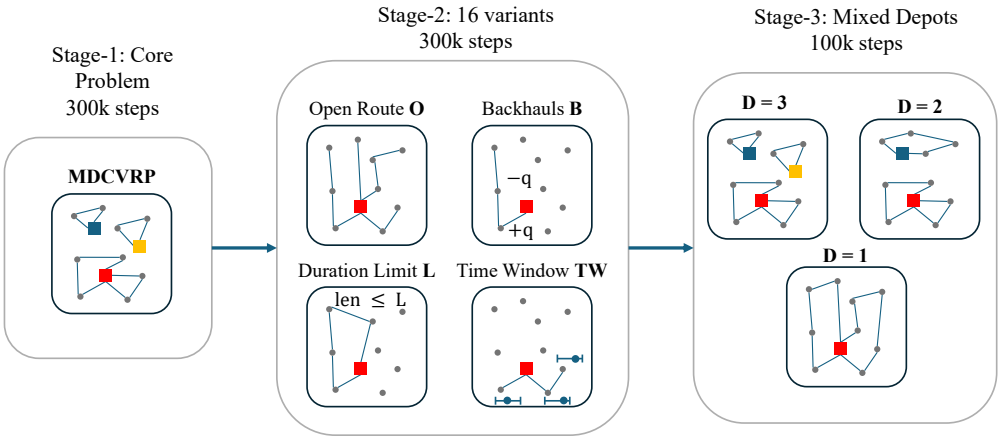


Figure 2: The unified three-stage curriculum. The same 5,074,656-parameter architecture is used from the first training step: 300k steps on the core MDCVRP with $D = 3$, 300k steps over all 16 operational variants with $D = 3$, and a 100k continuation in which both the variant and depot count $D \in \{1, 2, 3\}$ are sampled. Model weights are warm-continued between stages, while the optimizer and learning-rate schedule are restarted at each stage.

4.4 Unified Curriculum for Sixteen Variants

The final generalist covers all 16 combinations of open routes (O), mixed backhauls (B), route-length limits (L), and time windows (TW) (Liu et al., 2024; Zhou et al., 2024; Corrêa et al., 2026a). As shown in Fig. 2, one fixed architecture is trained for 300k steps on MDCVRP with $D = 3, N = 100$, continued for 300k steps with the 16 variants sampled uniformly, and then continued for 100k steps while sampling both the variant and $D \in \{1, 2, 3\}$. All input and state features exist from stage 1 and inactive attributes are zeroed, so only the training distribution changes.

Generalist batches contain 64 unaugmented instances and use $K = N + D - 1$ POMO starts with their mean reward as the within-instance baseline. Stages 1–2 use Adam with learning rate 10^{-4}, 5k-step warmup, and cosine decay; stage 3 uses 2×10^{-5} and a 2k-step warmup. Because depot tokens share one encoder, changing D does not change the network dimensions. The resulting checkpoint, *GenRoute-DM*, supports all 16 variants and $D \in \{1, 2, 3\}$ without depot- or variant-specific fine-tuning.

5 Experiments

Models. *GenRoute* denotes the refreshed stage-1 MDCVRP checkpoint trained at $D = 3, N = 100$. *GenRoute-Blind* is the corresponding depot-blind Stage-1 reference, in which the active-depot context is replaced by zeros. Both use the same Stage-1 training task, optimization recipe, budget, and random seed. *GenRoute-16* denotes the checkpoint after the 16-variant stage at $D = 3$, and *GenRoute-DM* denotes its final continuation across $D \in \{1, 2, 3\}$.

Data and metrics. We use controlled synthetic pools to isolate changes in routing variant, depot count, and problem size; RouteFinder's released suites provide standardized shared instances and references for comparison with multi-variant neural baselines; and CVRPLIB tests transfer to an established heterogeneous benchmark outside the training generator (Appendix F). Unless otherwise stated, synthetic instances have coordinates sampled uniformly from $[0, 1]^2$, integer customer demands sampled from $\{1, \ldots, 9\}$, and vehicle capacity $Q = 50$. Training uses $N = 100$; Stage 1 uses $D = 3$, while the final generalist is trained with $D \in \{1, 2, 3\}$. Test-time values of N and D are stated with each experiment. External instances are translated and uniformly scaled to fit within the unit square while preserving their aspect ratio, and demands are rescaled to the training capacity. Reconstructed route costs are converted back to the native instance scale.

We report $\text{Gap}(\%) = 100(L_{\text{model}} - L_{\text{ref}})/L_{\text{ref}}$. Percentage gap is scale-free and therefore permits comparison across variants and instance sizes with different objective magnitudes; for grouped syn-

Variant (reference)	MVMoE	MTPOMO	RF-EAL†	FiLMMeD	GR-DM
MDCVRP (11.89)	***12.13 (2.04)***	*12.29 (3.38)*	12.96 (9.14)	12.33 (3.71)	12.29 (3.35)
MDOVRP (7.97)	*8.39 (5.22)*	*8.40 (5.35)*	8.64 (8.37)	8.39 (5.27)	**8.32 (4.38)**
MDVRPL (11.90)	***12.13 (1.95)***	*12.29 (3.30)*	12.99 (9.31)	*12.22 (2.71)*	12.31 (3.47)
MDVRPTW (19.33)	*22.02 (13.94)*	*20.61 (6.64)*	21.14 (9.43)	*20.53 (6.23)*	**20.29 (4.99)**
MDOVRPL (7.97)	*8.41 (5.47)*	*8.43 (5.72)*	8.64 (8.33)	*8.36 (4.84)*	**8.32 (4.37)**
MDOVRPTW (13.00)	*14.40 (10.79)*	*14.21 (9.33)*	13.81 (6.24)	*13.54 (4.17)*	**13.44 (3.37)**
MDVRPLTW (19.35)	*21.93 (13.33)*	*20.50 (5.94)*	21.18 (9.47)	*20.44 (5.63)*	**20.33 (5.05)**
MDOVRPLTW (13.00)	*14.32 (10.17)*	*14.13 (8.71)*	13.81 (6.24)	*13.47 (3.63)*	**13.43 (3.36)**
MDVRPMB (10.68)	*11.22 (5.04)*	*11.20 (4.85)*	12.70 (18.98)	*11.19 (4.76)*	**11.07 (3.67)**
MDOVRPMB (7.66)	*8.50 (10.95)*	*8.48 (10.68)*	8.53 (11.35)	*8.18 (6.77)*	**7.92 (3.41)**
MDVRPMBL (10.71)	*11.17 (4.34)*	*11.15 (4.15)*	12.68 (18.56)	*11.13 (3.97)*	**11.12 (3.87)**
MDVRPMBTW (19.29)	*24.41 (26.57)*	*22.43 (16.31)*	21.41 (11.05)	*22.05 (14.34)*	**20.24 (4.95)**
MDOVRPMBL (7.66)	*8.51 (11.07)*	*8.48 (10.68)*	8.54 (11.43)	*8.19 (6.90)*	**7.92 (3.41)**
MDOVRPMBTW (12.96)	*15.69 (21.03)*	*15.10 (16.47)*	13.93 (7.41)	*14.38 (10.92)*	**13.39 (3.29)**
MDVRPMBLTW (19.31)	*24.34 (26.05)*	*22.39 (15.95)*	21.44 (11.09)	*22.02 (14.03)*	**20.27 (4.99)**
MDOVRPMBLTW (12.96)	*15.59 (20.25)*	*15.04 (16.01)*	13.91 (7.32)	*14.31 (10.38)*	**13.39 (3.28)**
Time (s / instance)	–	–	–	0.32	0.061

Table 1: RouteFinder's multi-depot 16-variant suite (D=3, N=100; 1,000 instances per variant). Cells give mean objective and gap to the listed 20 s HGS reference in parentheses (Berto et al., 2025). GR-DM uses augmentation 8 and greedy decoding. Upright values use the same released instances; †RF-EAL is from RouteFinder's Tables 2 and 11. Italic values are published results on independent draws (Liu et al., 2024; Zhou et al., 2024; Corrêa et al., 2026a). Boldface is descriptive, not a paired superiority claim. The last row is wall-clock inference time per instance on one H200 under each model's standard protocol (Appendix D); the quoted systems were not executed here.

thetic and RouteFinder results, we compute the gap between mean model and reference objectives and then average row-wise gaps for macro results; CVRPLIB instead averages its 14 instance-wise gaps. Synthetic references are produced by HGS through PyVRP (Vidal, 2022; Wouda et al., 2024); the time budget used to produce each reference is stated in the corresponding table. For external datasets, we report feasibility only when reconstructed routes have been checked against all constraints of the evaluated formulation.

Baselines and comparison protocol. For MDCVRP, we retrain MD-MTA on the matched synthetic distribution using its complete inference protocol (Li et al., 2024). We evaluate the official DeepMDV checkpoints (Nasehi et al., 2024) and the released FiLMMeD model and code (Corrêa et al., 2026a). We also use released RouteFinder and MoSES checkpoints where they support the evaluated problem (Berto et al., 2025; Pan et al., 2025). Results produced by executable models on identical instances are treated as paired comparisons. Published values evaluated on different random instances are shown only as contextual results: they are visually separated and are not used for boldface rankings, win counts, or statistical superiority claims. The primary paired 16-variant comparison uses 100 held-out instances per variant, unaugmented greedy inference for both models, and independent per-instance evaluation. On one H200 with one instance per pass, which the generalist's normalization requires, GenRoute-DM decodes an instance in 0.061 s (816 greedy rollouts) and MD-MTA in 0.076 s (1,020 rollouts); at ten instances per pass, Stage-1 GenRoute takes 0.011 s and MD-MTA 0.017 s, and FiLMMeD takes 0.32 s (Appendix D).

Active-depot conditioning (Fig. 3). GenRoute obtains a mean gap of 2.26%, whereas the depot-blind Stage-1 control obtains 2.69%. The depot-blind control produces the higher cost on 691 of the 1,000 paired instances, a mean-gap difference of 0.43 percentage points. Both runs use the same Stage-1 task, optimization recipe, budget, and random seed; we therefore use this comparison as evidence for active-depot conditioning, but not as a strict one-variable architecture ablation.

Distance bias. A Stage-1 twin trained without the bias (same data, recipe, budget, and seed) reaches a lower gap in distribution, 1.79% against 2.26% on the 1,000-instance MDCVRP pool, but higher gaps under shift: 10.58% against 8.45% on the CVRPLIB layouts, and 4.31% against 3.35% at N=500 and 5.73% against 3.82% at N=1,000 on fresh MDCVRP pools, where the biased model has the lower cost on 93 of 100 and 50 of 50 paired instances (Appendix C).

Multi-depot variants (Table 1). On RouteFinder's released multi-depot sets (16 variants $\times$ 1,000 instances, with every gap measured against 20 s PyVRP references), GenRoute-DM has a 3.95%

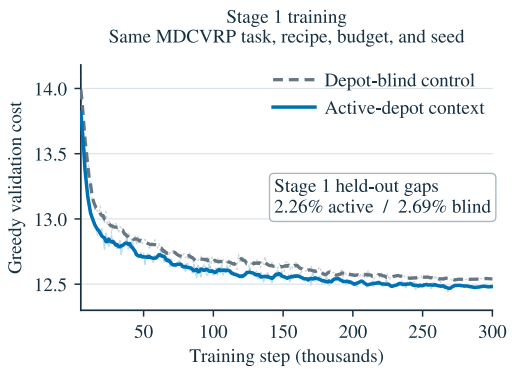


Figure 3: Active-depot comparison on 1,000 MDCVRP instances (D=3, N=100). GenRoute obtains a 2.26% mean gap, while the controlled depot-blind Stage-1 reference obtains 2.69%.

	Gap to HGS–PyVRP (2 s), %								
Variant	GR-DM^m	FiLMMeD^z	MoS-C	MoS-R	RF-TE	RF-MoE	MVMoE	RF-POMO	MTPOMO
CVRP	1.75	3.74	1.32	**1.14**	1.46	1.59	1.66	1.79	1.95
OVRP	5.08	9.43	3.68	**3.51**	4.07	4.26	4.65	4.68	4.99
VRPL	2.36	4.39	1.65	**1.51**	1.81	1.93	2.12	2.16	2.44
VRPTW	4.44	5.59	**2.39**	2.85	3.19	3.53	3.81	3.59	3.89
OVRPL	5.11	9.16	3.67	**3.49**	4.07	4.27	4.73	4.68	5.04
OVRPTW	3.45	5.41	**1.73**	2.11	2.37	2.72	2.89	2.75	2.92
VRPLTW	5.01	6.32	**2.85**	3.34	3.63	3.95	4.31	3.99	4.41
OVRPLTW	3.47	5.39	**1.74**	2.11	2.37	2.71	2.91	2.75	2.92
Mean	3.83	6.18	**2.38**	2.51	2.87	3.12	3.39	3.30	3.57

Table 2: Mean gaps to 2 s HGS–PyVRP references on RouteFinder's released single-depot sets (Berto et al., 2025) (D=1, N=100; 1,000 instances per variant). All models use augmentation 8, greedy decoding, and full multi-start budgets. ^{m}Training mixes $D \in \{1, 2, 3\}$; ^{z}zero-shot transfer from multi-depot training. MoS-C/R use the CaDA/RouteFinder backbones (Li et al., 2025; Berto et al., 2025). Boldface marks the lowest row value.

mean across all 16,000 solutions and is best in 14 of 16 rows; the reported alternatives average 9.0–11.8%. MD-MTA, which is designed for MDCVRP, measures 3.41%. The largest margins arise on the variants that combine mixed backhauls with time windows, where GenRoute-DM is 4 to 6 points below the next entry. These are the variants whose feasibility depends on the most state, the collected backhaul load and the elapsed time, both of which the decoder context carries explicitly, and on which the alternatives were not trained under the mixed-backhaul convention (Appendix F).

Single-depot suite (Table 2). On the eight variants used to train the single-depot systems, evaluated unchanged at D=1 under one protocol, their averages vary from 2.4–3.6%. GenRoute-DM reaches 3.8%. FiLMMeD and RF-EAL results are reported in the table. The eight mixed-backhaul variants are the mirrored situation: our checkpoint trained on them (in multi-depot form) and the single-depot systems did not, so that comparison measures training exposure rather than model quality; it is reported in the technical appendix under "Mixed-Backhaul Single-Depot Variants." GenRoute-DM is 0.3 percentage points above the upper end of the single-depot systems' range on their trained variants (3.8% vs. 2.4–3.6%); those systems are single-depot-only architectures, while the other multi-depot-capable entries trail it (RF-EAL 4.9%, FiLMMeD 6.2%). Its distance to the best single-depot system is smallest on CVRP and VRPL and largest on the open-route and time-window variants, up to 2.2 points on VRPLTW; the single-depot systems were trained on exactly these eight variants at D=1, which GenRoute-DM sees only as one of three depot counts in its final stage, so the gap measures training exposure as much as architecture.

Depot-count and problem-size generalization. Across $D \in \{2, \ldots, 5\}$ and $N \in \{50, 100, 200\}$, GenRoute-DM obtains the lowest gap in 9 of the 12 settings. MD-MTA leads at $D = 5, N = 50$ and at $D \in \{4, 5\}, N = 100$. GenRoute-DM's margin grows with N and it leads every depot count at N=200 by 1.2 to 1.8 points; MD-MTA leads only in extrapolation cells at $N \leq 100$, where

	N=50			N=100			N=200		
D	GR-DM	FiLM	MD-MTA	GR-DM	FiLM	MD-MTA	GR-DM	FiLM	MD-MTA
2	**2.93**	3.67	4.89	**1.79**	2.43	1.95	**1.62**	3.96	3.42
3	**3.88**	4.76	5.52	**2.33**	2.79	2.35	**1.68**	3.49	3.50
4*	**4.61**	6.05	5.39	2.86	3.23	**2.82**	**2.07**	3.56	3.70
5*	7.39	8.74	**6.13**	3.50	4.23	**3.06**	**2.83**	4.02	4.37
Time (s)	0.033	0.21	0.047	0.062	0.56	0.078	0.126	2.23	0.170

Table 3: Mean gaps to 2 s HGS–PyVRP on 200 matched MDCVRP instances per setting. Every GenRoute instance is decoded independently (batch size one). All models train at N=100; GenRoute-DM mixes $D \in \{1, 2, 3\}$, while FiLMMeD and MD-MTA train at D=3. All extrapolate for $D \in \{4, 5\}^*$. Boldface marks the lowest gap. The last row is wall-clock seconds per instance on one H200, averaged over D, with every model decoding one instance per pass (Appendix D).

every model is outside its training depot range, so the ordering there is not evidence about either architecture.

One generalist for 16 variants. The expanded generalist is compared head-to-head against FiLMMeD (Corrêa et al., 2026a)—the strongest released multi-depot generalist among the baselines evaluated here—under its own conventions (verified line-by-line), on 100 paired instances per variant from an untouched suite drawn after all design decisions were frozen and held out of training and checkpoint selection (appendix). Every instance is evaluated independently so that its prediction cannot depend on unrelated test instances through the batch-statistics normalization. On the untouched suite, GenRoute-DM achieves an aggregate mean objective of 13.754, compared with 14.074 for FiLMMeD, corresponding to a 2.3% reduction; averaged per variant, the relative reduction is 2.1% (95% confidence interval 1.9 to 2.2%, paired bootstrap over instances). GenRoute-DM obtains the lower mean objective in 9 of 16 variants and the lower cost on 1,083 of the 1,600 paired instances. The gain concentrates on the eight mixed-backhaul variants, which are 1.0 to 6.6% lower, while the other eight variants lie within 1.1% either way.

6 Limitations

As a pure constructor without local search, GenRoute does not always match classical or hybrid methods. More importantly, its broad multi-depot coverage does not give the best result on every single-depot problem: models trained specifically for $D = 1$ remain stronger on CVRPLIB and on the single-depot variants used in their training. Generalization also weakens at the largest depot-count shift, D=5, N=50, outside the training range $D \in \{1, 2, 3\}$. These results expose a coverage–specialization trade-off: one checkpoint spans many settings, whereas dedicated $D = 1$ training remains preferable when only single-depot performance matters. Improving single-depot specialization without losing cross-variant coverage is therefore the main direction for future work.

7 Conclusion

GenRoute shows that multi-depot construction can be learned without separate depot encoders or explicit customer allocation. Its shared distance-biased encoder represents all depots, customers, and attributes, while the active-depot-conditioned decoder allocates customers and constructs depot-owned routes within one action sequence. In the controlled Stage-1 comparison, GenRoute obtains a 2.26% MDCVRP gap versus 2.69% for the depot-blind reference. The curriculum extends the same architecture to 16 operational variants and $D \in \{1, 2, 3\}$. GenRoute-DM achieves a 3.95% macro gap across the multi-depot suite and the lowest gap in 9 of 12 depot-count and problem-size settings. In the paired 1,600-instance evaluation, it reduces the aggregate mean objective by 2.3% and produces the lower-cost solution on 1,083 instances. These results demonstrate broad coverage from one constructive checkpoint, while improving single-depot specialization and unseen-depot extrapolation remain important directions. This coverage is achieved without variant-specific fine-tuning or local search, while the learned distance bias remains part of the fixed architecture throughout the curriculum.

AI USE STATEMENT

In this work, we used generative AI tools to implement methods, to provide feedback on the experimental protocol, and to compute and summarize results. We did not use generative AI tools to generate synthetic data sets, to help develop theoretical models or conceptual frameworks, to formulate mathematical claims, to provide critical ingredients for proving mathematical claims, to assist in the writing of proofs, to propose or refine hypotheses, to assist with translation, or to support qualitative and thematic data analysis. We also used them to draft and edit text, create some of the figures, and check references. We take responsibility for the final content of this work.

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)*, 2017.

Federico Berto, Chuanbo Hua, Nayeli Gast Zepeda, André Hottung, Niels A. Wouda, Leon Lan, Junyoung Park, Kevin Tierney, and Jinkyoo Park. RouteFinder: Towards foundation models for vehicle routing problems. *Transactions on Machine Learning Research*, 2025.

Jean-François Cordeau, Michel Gendreau, and Gilbert Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30(2):105–119, 1997.

Arthur Corrêa, Paulo Nascimento, and Samuel Moniz. FiLMMeD: Feature-wise linear modulation for cross-problem multi-depot vehicle routing. *arXiv preprint arXiv:2604.28102*, 2026a.

Arthur Corrêa, Cristóvão Silva, Liming Xu, Alexandra Brintrup, and Samuel Moniz. TuneNSearch: A hybrid transfer learning and local search approach for solving vehicle routing problems. *Computers & Operations Research*, 190:107433, 2026b. doi: 10.1016/j.cor.2026.107433.

Chaitanya K. Joshi, Quentin Cappart, Louis-Martin Rousseau, and Thomas Laurent. Learning the travelling salesperson problem requires rethinking generalization. *Constraints*, 27(1):70–98, 2022. doi: 10.1007/s10601-022-09327-y.

Minsu Kim, Junyoung Park, and Jinkyoo Park. Sym-NCO: Leveraging symmetricity for neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 35, 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, 2020.

Yeong-Dae Kwon, Jinho Choo, Iljoo Yoon, Minah Park, Duwon Park, and Youngjune Gwon. Matrix encoding networks for neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

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, 2025.

Jinqi Li, Bing Tian Dai, Yunyun Niu, Jianhua Xiao, and Yaoxin Wu. Multi-type attention for solving multi-depot vehicle routing problems. *IEEE Transactions on Intelligent Transportation Systems*, 25(11):17831–17840, 2024. doi: 10.1109/TITS.2024.3413077.

Fei Liu, Xi Lin, Zhenkun Wang, Qingfu Zhang, Xialiang Tong, and Mingxuan Yuan. Multi-task learning for routing problem with cross-problem zero-shot generalization. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1898–1908, 2024. doi: 10.1145/3637528.3672040.

Saeed Nasehi, Farhana Choudhury, Egemen Tanin, and Majid Sarvi. DeepMDV: Global spatial matching for multi-depot vehicle routing problems. *arXiv preprint arXiv:2411.17080*, 2024.

Mohammadreza Nazari, Afshin Oroojlooy, Lawrence Snyder, and Martin Takáč. Reinforcement learning for solving the vehicle routing problem. In *Advances in Neural Information Processing Systems*, volume 31, 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, 2025.

Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, and Anand Subramanian. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858, 2017. doi: 10.1016/j.ejor.2016.08.012.

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.

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.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Yang Wang, Ya-Hui Jia, Wei-Neng Chen, and Yi Mei. Distance-aware attention reshaping: Enhance generalization of neural solver for large-scale vehicle routing problems. *arXiv preprint arXiv:2401.06979*, 2024. doi: 10.48550/arXiv.2401.06979.

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

Niels A. Wouda, Leon Lan, and Wouter Kool. PyVRP: A high-performance VRP solver package. *INFORMS Journal on Computing*, 36(4):943–955, 2024. doi: 10.1287/ijoc.2023.0055.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Zhi Zheng, Shunyu Yao, Zhenkun Wang, Xialiang Tong, Mingxuan Yuan, and Ke Tang. DPN: Decoupling partition and navigation for neural solvers of min-max vehicle routing problems. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61559–61592. PMLR, 2024. URL `https://proceedings.mlr.press/v235/zheng24m.html`.

Changliang Zhou, Canhong Yu, Shunyu Yao, Xi Lin, Zhenkun Wang, Yu Zhou, and Qingfu Zhang. URS: A unified neural routing solver for cross-problem zero-shot generalization. In *International Conference on Machine Learning*, 2026.

Jianan Zhou, Zhiguang Cao, Yaoxin Wu, Wen Song, Yining Ma, Jie Zhang, and Xu Chi. MVMoE: 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, 2024.

TECHNICAL APPENDIX

We use the main paper's model names. *GenRoute* is the Stage 1 MDCVRP checkpoint at $D{=}3$, $N{=}100$, and *GenRoute-Blind* is the controlled depot-blind Stage 1 reference. *GenRoute-16* is trained across all 16 variants at $D{=}3$; *GenRoute-DM* is its final continuation across $D \in \{1, 2, 3\}$. All three GenRoute checkpoints use the same distance-biased architecture; only the training distribution changes across curriculum stages.

Aspect	GenRoute	MD-MTA	TuneNSearch
Node encoding	One shared attention stream over all $D + N$ depot and customer tokens	Separate depot, customer, and all-node streams with multi-type attention	Transformer encoder augmented with edge-aware graph attention
Geometry	Learned per-head bias from pairwise Euclidean distances	Coordinates represented through the node embeddings	Pairwise distances represented through edge-aware attention
Decoder	Pointer decoder conditioned explicitly on the active depot and resource state	Multi-type decoder attention over the separate embedding streams	Constructive pointer decoder
Training	One fixed architecture trained through a curriculum over routing variants and depot counts	Reinforcement learning from scratch for MDVRP and MDOVRP	MDVRP pretraining followed by target-specific fine-tuning
Post-processing	None; pure construction	None; pure construction	Local-search refinement after neural construction
Reported scope	One checkpoint for 16 variants and $D \in \{1, 2, 3\}$	Separate evaluations on MDVRP and MDOVRP	Multiple VRP variants through target-specific fine-tuning

Table A1: Architectural comparison with two representative neural multi-depot constructors discussed in the main paper. The table summarizes model and inference design rather than providing a protocol-matched performance ranking. Pure construction means that the decoded solution is not refined by a separate local-search stage.

A Architecture and Relation to POMO

Comparison with Multi-Depot Constructors. Table A1 compares GenRoute with MD-MTA and TuneNSearch, two representative neural multi-depot constructors discussed in the main paper (Li et al., 2024; Corrêa et al., 2026b). GenRoute represents all depots and customers in one shared attention stream and provides the encoded active depot directly to the decoder. MD-MTA instead combines separate depot, customer, and all-node embedding streams through multi-type attention. TuneNSearch augments a Transformer encoder with edge-aware graph attention and uses transfer learning followed by local search.

These systems therefore differ in both model design and inference protocol. GenRoute and MD-MTA are pure constructors, whereas TuneNSearch refines its neural solution using local search.

Relation to POMO. GenRoute follows the Attention Model and POMO construction framework (Kool et al., 2019; Kwon et al., 2020). It retains a shared encoder–pointer-decoder structure, POMO multi-start decoding, and a reward baseline shared across starts. Table A2 separates these inherited elements from capacity changes and from the extensions needed for multi-depot and multi-variant routing.

GenRoute's central multi-depot extension is the encoded active depot in the decoder context; with the enlarged action space, feasibility mask, and start set, it lets one decoder construct routes for multiple depots. Every encoder layer additionally incorporates a learned per-head distance bias, while the larger embedding increases model capacity.

B Variant Definitions and Construction Details

The main paper defines the base MDCVRP objective and construction MDP. This section gives the additional generation, transition, and masking conventions needed to reproduce the complete 16-variant environment. The variant family and standard generator conventions follow prior multi-variant routing work (Liu et al., 2024; Zhou et al., 2024; Berto et al., 2025; Corrêa et al., 2026a); we state the exact conventions used here because small implementation choices change the resulting environment.

Component	POMO	GenRoute	Relation
Input representation	One depot and N customers, with separate depot and customer projections	D depots and N customers in one shared token sequence; the complete variant feature set is fixed from the first training stage	Extended
Encoder capacity	$L = 6, E = 128, H = 8$, and feed-forward dimension 512	$L = 6$, $E = 256$, $H = 16$, and feed-forward dimension 1024	Widened
Normalization	Instance normalization	Batch normalization; GenRoute-16 and GenRoute-DM are evaluated one instance at a time	Changed
Encoder attention	Standard scaled dot-product attention	Standard attention plus a learned per-head distance bias	New
Decoder context	Last-node embedding and remaining vehicle capacity	Last-node embedding, encoded active depot, and resource state; the generalist appends the states required by the active constraints	Extended
Action space and mask	One depot and N customers, with visited-customer and capacity masks	D depots and N customers, with own-depot route closure and variant-specific feasibility masks	Extended
Multi-start set	$K = N$ customer starts	$K = N + D - 1$: all customer starts and depot starts $2, \ldots, D$	Extended
Training baseline	Mean reward over the K starts of each instance	Mean reward over the complete $K = N + D - 1$ start set of each instance	Adapted
Pointer head	Clipped single-head pointer followed by a masked softmax	Same pointer and masked-softmax construction	Inherited
Inference	Greedy selection over augmented customer-start rollouts	Greedy selection over $A(N + D - 1)$ augmentation–start rollouts	Adapted
Variant capability	Single-task formulation	All variant inputs are present from the first stage and inactive inputs are zeroed; the same $5{,}074{,}656$ parameters are used throughout the curriculum	Extended

Table A2: Component-level relation between canonical POMO and GenRoute. Inherited components retain the same function, widening changes capacity only, and adapted or extended components support multi-depot or multi-variant operation. The principal multi-depot extension is the active-depot representation in the decoder context; the learned distance bias augments every encoder layer with pairwise geometric information.

B.1 Variant Family and Instance Generation

Unless otherwise stated, synthetic depot and customer coordinates are sampled independently from $U[0,1]^2$, customer-demand magnitudes are sampled from $U\{1,\ldots,9\}$, and vehicle capacity is $Q = 50$. The formulation permits an unlimited number of routes per depot. Every closed route returns to the depot that owns it, and setting $D = 1$ recovers the corresponding single-depot problem.

The generalist covers every combination of four binary attributes. Let

$$r_i = \max_{\delta \in V_{\mathrm{dep}}} d_{\delta i}, \qquad r_{\max} = \max_i r_i.$$

For a generated training batch, the same randomly selected set of backhaul customer indices is used across the instances in that batch, while their coordinates and demand magnitudes remain independently sampled. Attributes that are inactive for a variant are represented by zero-valued tensors.

The lower bound $\bar{\rho} \geq 2r_{\max}$ ensures that a fresh closed route from any depot can visit any single customer and return within the route-length limit. The time-window construction similarly ensures

$$l_i + s_i + r_i \leq H,$$

so a customer served anywhere within its window can return to any depot before the horizon.

Attribute	Instance generation	Route semantics
Open routes (O)	The instance flag is $o = 1$.	The return edge from the final customer of every route to its owning depot is omitted from the objective, the route-length calculation, and the return-horizon check.
Mixed backhauls (B)	Exactly $\lfloor 0.2N \rfloor$ customer indices are selected without replacement. Their demands are negated while retaining magnitudes in $\{1, \ldots, 9\}$.	Deliveries and pickups may be interleaved. Separate counters track delivered linehaul demand and collected backhaul demand. B therefore denotes the mixed-backhaul convention; no linehaul-before-backhaul precedence is imposed.
Route-length limit (L)	Each instance receives $\bar{\rho} \sim U[2r_{\max}, 3]$.	The travel distance of each route may not exceed $\bar{\rho}$. Waiting and service time do not contribute to this limit. The return edge is included for closed routes and omitted for open routes.
Time windows (TW)	With horizon $H = 4.6$, sample $s_i \sim U(0.15, 0.18)$ and window width $w_i \sim U(0.18, 0.20)$, followed by $e_i \sim U[r_i, H - s_i - w_i - r_i]$ and $l_i = e_i + w_i$.	A vehicle may wait when it arrives before e_i. Service must begin no later than l_i; a closed route must also return to its active depot before H. Vehicle speed is one.

Table A3: The four binary attributes defining the 16-variant family. Any subset of $\{\mathrm{O}, \mathrm{B}, \mathrm{L}, \mathrm{TW}\}$ may be active. The letter B in all result tables denotes mixed backhauls.

B.2 Construction State and Transitions

For the complete variant family, the construction state is

$$s_t = \left(a_{t-1}, \delta_t, u_t^{\mathrm{lh}}, u_t^{\mathrm{bh}}, \xi_t, \rho_t, U_t\right),$$

where a_{t-1} is the current node, δ_t is the active depot that owns the current route, u_t^{lh} is the cumulative linehaul demand served on that route, u_t^{bh} is the cumulative backhaul demand collected, ξ_t is service-completion time, ρ_t is the partial route length, and U_t is the unvisited-customer set. In the notation of the main paper,

$$\ell_t = Q - u_t^{\mathrm{lh}}, \qquad \beta_t = u_t^{\mathrm{bh}}.$$

For a selected customer j, define

$$\Delta_{tj} = d(a_{t-1}, j).$$

The transition adds Δ_{tj} to the objective and to ρ_t. When time windows are active, time is updated as

$$\xi_{t+1} = \max(\xi_t + \Delta_{tj}, e_j) + s_j.$$

Otherwise, ξ_t remains zero. The appropriate linehaul or backhaul counter is increased, and j is removed from U_t.

Selecting depot d acts as both a route delimiter and a next-route selector. If the current route is closed, the edge from the current customer to its owning depot δ_t is added to the objective. This edge is omitted for an open route. The route-specific state is then reset, and the selected depot becomes the owner of the next route:

$$\delta_{t+1} = d, \qquad u_{t+1}^{\mathrm{lh}} = u_{t+1}^{\mathrm{bh}} = \xi_{t+1} = \rho_{t+1} = 0.$$

No travel cost is assigned from δ_t to d, because the depot action starts a new vehicle route rather than physically relocating the previous vehicle. When all customers have been visited, the final route is closed using the same convention.

B.3 Feasibility Mask

Customer j is selectable only if it is unvisited and every active constraint is satisfied. For a linehaul customer with $q_j \geq 0$, the implemented capacity conditions are

$$u_t^{\mathrm{lh}} + q_j \leq Q, \qquad q_j \leq Q - u_t^{\mathrm{bh}}.$$

For a backhaul customer with $q_j < 0$, the condition is

$$u_t^{\mathrm{bh}} + |q_j| \leq Q.$$

These conditions permit deliveries and pickups to be interleaved.

For a customer with a time window, service must begin within the window:

$$\max(\xi_t + \Delta_{tj}, e_j) \le l_j.$$

For a closed route, the mask additionally guarantees completion of service and return to the active depot before the horizon:

$$\max(\xi_t + \Delta_{tj}, e_j) + s_j + d(j, \delta_t) \le H.$$

When a route-length limit is active, the corresponding check is

$$\rho_t + \Delta_{tj} + \begin{cases} d(j, \delta_t), & \text{closed route,} \\ 0, & \text{open route} \end{cases} \le \bar{\rho}.$$

All comparisons use a numerical tolerance of 10^{-6}.

All D depot actions are available while the current node is a customer. They are all masked while the current node is a depot, preventing consecutive depot actions under the normal mask.

For the synthetic generator, these one-step masks are sufficient without lookahead. If the current route cannot accept another customer, it can be closed and a fresh route can serve any remaining customer: its demand is below Q, the route-length lower bound permits a round trip from any depot, and the time-window construction permits service and return before the horizon. Thus, an earlier action cannot strand a customer on the generated training and fresh-test distributions.

The implementation records a diagnostic if an unfinished rollout has no feasible action. This diagnostic is not treated as a general feasibility guarantee for arbitrary external data. When evaluating released external variant suites, incomplete rollouts are excluded before selecting the lowest-cost solution. Feasibility is claimed only for experiments in which the reconstructed routes were checked against the evaluated formulation.

B.4 Multi-Start Construction

For an instance with N customers and D depots, the model constructs

$$K = N + D - 1$$

parallel rollouts. The first $D - 1$ are initialized at depots $2, \ldots, D$. The remaining N rollouts each force a different customer as the first action from depot 1. A forced customer action pays the edge from depot 1, updates the route state, and is excluded from the policy-gradient likelihood. The synthetic generation bounds above guarantee that these forced actions are feasible.

After initialization, each rollout repeatedly applies the feasibility mask, selects an action by sampling during training or by argmax during inference, and applies the corresponding transition. An episode contains at most $2N$ actions. At inference, augmentation 8 returns the least-cost feasible solution among the $8K$ combinations of geometric transformation and forced start. Experiments using unaugmented inference instead use only the K starts and are identified explicitly in their captions.

B.5 Encoded Features

Depot tokens contain only their coordinates. The final generalist represents customer i by

$$[p_i;\ |q_i|;\ e_i;\ l_i;\ s_i;\ b_i;\ o]\,, \qquad b_i = \mathbb{1}[q_i < 0].$$

Thus, the environment retains signed demands, but the network receives the demand magnitude together with an explicit backhaul indicator.

At each construction step, the decoder context contains

$$\left[h_{a_{t-1}};\ h_{\delta_t};\ Q - u_t^{\mathrm{lh}};\ \xi_t;\ \rho_t;\ \bar{\rho};\ u_t^{\mathrm{bh}};\ o\right].$$

The current route length ρ_t is always updated and supplied. When the corresponding attributes are inactive, e_i, l_i, s_i, b_i, ξ_t, $\bar{\rho}$, u_t^{bh}, and o are zero. Features are supplied in their raw generator units without additional feature normalization.

The formulation and generator accept arbitrary D and N. Curriculum stages 1–2 use $D = 3, N = 100$, while the final stage samples $D \in \{1, 2, 3\}$ with $N = 100$. Test-time depot and customer counts are stated with each corresponding experiment.

Configuration	Training distribution	Training steps	Batch	A_{train}	Baseline	Initial LR
GenRoute / GenRoute-Blind	MDCVRP, $D = 3$	300k	64	1	K	10^{-4}
GenRoute-16	16 variants, $D = 3$	300k	64	1	K	10^{-4}
GenRoute-DM	16 variants, $D \in \{1, 2, 3\}$	100k	64	1	K	2×10^{-5}

Table A4: Training configurations used in the paper. All configurations use $N = 100$, Adam with $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, zero weight decay, no gradient clipping, and $K = N + D - 1$ forced starts. A_{train} is the number of geometric copies used during training. The first two curriculum stages use a 5k-step linear warmup followed by cosine decay; the mixed-depot stage uses a 2k-step warmup. Checkpoints are selected using fixed validation pools. Model weights are continued between curriculum stages, whereas the optimizer and learning-rate schedule are restarted.

N	n	HGS	GenRoute	Bias-free twin	Paired
200	100	20.94	$+1.74\%$	$+1.89\%$	52/100
500	100	45.56	$+3.35\%$	$+4.31\%$	93/100
1,000	50	86.16	$+3.82\%$	$+5.73\%$	50/50

Table A5: Distance-bias twins under size shift. Fresh MDCVRP pools at D=3 (n instances per size, seed 20260730, unseen by both models); HGS is the mean cost of 2 s HGS–PyVRP references on the same instances, and gaps are relative to it; both twins use per-instance augmentation-8 greedy decoding with K=N+D−1 starts. "Paired" counts instances on which the distance-biased model has the lower cost. One seed per twin.

C Training and Evaluation Protocols

Implementation. The unified curriculum and refreshed evaluations use Python 3.9.25, PyTorch 2.1.2+cu121, CUDA 12.1, cuDNN 8.9.2, NumPy 1.26.4, SciPy 1.13.1, and PyVRP 0.9.1. Training is performed in full precision on one NVIDIA H200 NVL GPU with 139.8 GiB memory. The host runs Rocky Linux 9.8 with two AMD EPYC 9455 CPUs and 755.1 GiB RAM. HGS references are produced using PyVRP 0.9.1. The model uses six encoder layers, embedding dimension 256, 16 attention heads, feed-forward width 1024, and clipped pointer logits with $C = 10$. Each encoder layer augments dot-product attention with a learned per-head offset computed from $[d_{ij}, d_{ij}^2, \log(d_{ij}+\varepsilon), (d_{ij}+\varepsilon)^{-1}]$ by a $4 \to 64 \to 16$ MLP. The output projection is initialized to zero; across six layers, these modules add 8,160 parameters. They remain present throughout Stages 1–3 and at inference.

Training configurations. Table A4 summarizes the distinct training protocols. It covers the unified curriculum and the retained active-depot comparison.

Active-depot reference. GenRoute and GenRoute-Blind use the first configuration in Table A4: the same MDCVRP distribution, optimization settings, 300k-step budget, and random seed. In GenRoute-Blind, the encoded active-depot vector in the decoder context is replaced by zeros throughout both training and inference. The reference predates the otherwise inactive open-route input column in the refreshed Stage 1 architecture, so we treat the result as a controlled reference.

Distance-bias reference. A matched Stage-1 twin trained without the pairwise-distance bias (identical data, optimization settings, budget, and seed; the bias module's $8{,}160$ parameters are the only difference) quantifies the module's contribution. In distribution the bias-free twin is slightly stronger: on the 1,000-instance MDCVRP pool it reaches a 1.79% gap versus GenRoute's 2.26% (paired per-instance difference 0.056; the lower cost on 717 of $1{,}000$ instances). Under transfer to the CVRPLIB layouts of the main paper the ordering reverses: Stage-1 GenRoute reaches an 8.45% mean gap versus 10.58% for the bias-free twin, with the lower gap on 10 of 14 instances and the largest per-instance differences (5–7 percentage points) on the irregular layouts. The bias therefore trades a small in-distribution premium for robustness on shifted layouts. This reference uses one seed and, like the depot-blind reference, changes exactly one component against Stage 1. Under size shift the same comparison strengthens monotonically with the instance size (Table A5): the twins are statistically tied at N=200, while at N=1,000 the distance-biased model has the lower cost on every paired instance.

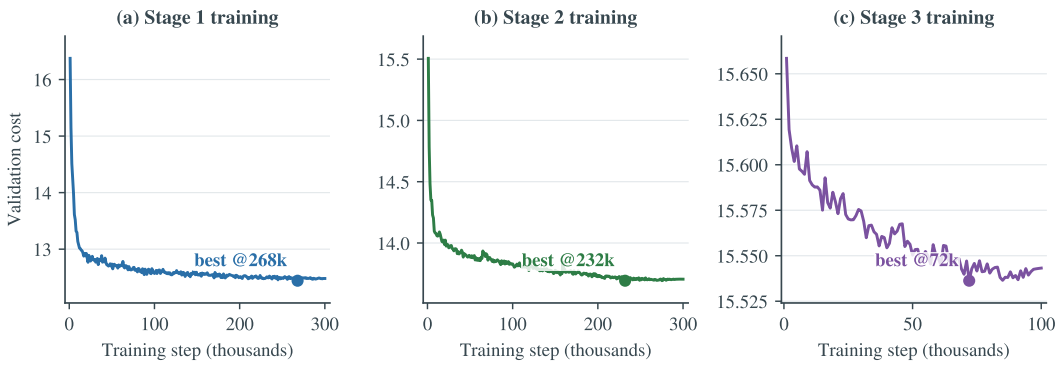


Figure A1: Validation-cost trajectories for the unified curriculum. (a) Stage 1 trains the core MDCVRP model at $D = 3$ and selects the 268k-step checkpoint loaded into stage 2. (b) Stage 2 trains across all 16 variants at $D = 3$ and selects the GenRoute-16 checkpoint at 232k steps. (c) Stage 3 samples both the variant and $D \in \{1, 2, 3\}$ and selects GenRoute-DM at 72k steps. The panels use different validation distributions, so their vertical scales are not directly comparable.

Unified curriculum. The same 5,074,656-parameter architecture is used throughout all three curriculum stages. Every customer attribute and decoder-state input is present from the first optimization step. Attributes that are inactive for a sampled problem are represented by zeros, but the network dimensions and parameterization do not change.

The core MDCVRP checkpoint is continued for 300k steps with the 16 variants sampled uniformly at $D = 3$, producing GenRoute-16. Its weights are then continued for 100k steps while both the problem variant and $D \in \{1, 2, 3\}$ are sampled, producing GenRoute-DM. Consequently, the curriculum changes only the training distribution; it does not expand, replace, or graft architectural components between stages.

Figure A1 shows the validation trajectories and the checkpoint selected at each curriculum stage.

Inference. Inference is greedy. For an instance with N customers and D depots, our models decode $K = N + D - 1$ forced starts. Unless a table states otherwise, we apply the eight dihedral transformations of the unit square and retain the lowest-cost feasible solution among the resulting $8K$ rollouts. The unaugmented paired 16-variant comparison instead uses $A = 1$ for both models.

GenRoute-16 and GenRoute-DM use batch normalization without stored running statistics. They are therefore evaluated one underlying instance at a time; the geometric copies and forced starts belonging to that instance may be batched together, but unrelated instances are not mixed in the same normalization batch.

Executable baselines are evaluated on the same instances using their released checkpoints and native greedy decoding protocols. Results quoted from the literature on different test samples are treated only as contextual results and are not included in boldface rankings or win counts.

Instances and scoring. Unless otherwise stated, synthetic instances use coordinates sampled uniformly from $[0, 1]^2$, integer customer demands from $\{1, \ldots, 9\}$, and vehicle capacity $Q = 50$. Training uses $N = 100$; the depot count used by each configuration is shown in Table A4.

External instances are translated and uniformly scaled to fit inside the unit square while preserving their aspect ratio. Demands are rescaled to preserve the original demand-to-capacity ratio. Constructed routes are then mapped back to the native coordinate and demand scales before scoring. CVRPLIB routes are scored under the official integer EUC_2D metric and checked against the original capacity constraints. RouteFinder evaluation uses its released instances without changing their problem attributes. The HGS time budget and evaluation-pool size are stated in each corresponding table caption.

Runs, randomness, and checkpoint selection. The reported GenRoute checkpoints come from one end-to-end three-stage training realization. Each stage is run once with seed 42; Stages 2 and 3 load the selected weights from the preceding stage while restarting Adam and the learning-rate

Model	A	Rollouts	Objective	Gap (%)
MD-MTA	1	102	**12.506**	**4.55**
GenRoute	1	102	12.547	4.90
MD-MTA	10	1,020	12.277	2.64
GenRoute	8	816	**12.232**	**2.26**

Table A6: Effect of inference-time augmentation on the same $1{,}000$ MDCVRP instances with $D{=}3$ and $N{=}100$. The HGS–PyVRP reference mean is 11.961. Here, A is the number of transformations and the number of rollouts is AK, with $K = 102$. The first block disables augmentation for both models; the second uses each model's standard augmentation protocol. Boldface marks the lower objective and gap within each block.

schedule. The reported depot-blind reference is one separate seed-42 run. At the start of every stage, Python's `random` and the PyTorch random-number generator are initialized to 42. On-policy training generation does not use NumPy, and deterministic CUDA kernels are not enforced. No systematic hyperparameter sweep was conducted: the POMO-style architecture and optimization recipe were fixed before the reported curriculum, and model selection was restricted to its checkpoints.

Validation is performed every 1,000 optimization steps on fixed pools. Stage 1 uses 32 MDCVRP instances at $D = 3$; Stage 2 uses 32 instances per variant at $D = 3$; and Stage 3 uses 32 instances per variant separately at $D = 1$ and $D = 3$. The $D = 3$ pools use PyTorch generator seed 49, and the additional Stage 3 $D = 1$ pools use seed 4207. For every instance, validation retains the lowest-cost greedy solution among the $K = N + D - 1$ starts and averages first within and then equally across validation groups. A checkpoint replaces the incumbent when this macro mean improves by more than 10^{-3}. The selected checkpoints occur at 268k, 232k, and 72k stage-local steps. Stages 1–2 use their complete 300k-step budgets; Stage 3 uses patience 80 validations and completes its 100k-step budget.

Reproducibility artifacts. The submitted anonymous code-and-data supplement contains the implementation, final checkpoints, machine-readable run configurations, preprocessing and evaluation code, frozen evaluation pools, per-instance outputs, aggregation scripts, and a SHA-256 manifest under the MIT license. The fresh paired suite contains 100 held-out instances for each of 16 variants at $D = 3, N = 100$ (generator seed 20260720; SHA-256 `bec1e7e7c135b59d37118629 04834807796a70707cdd20e8eb9ed90ad2284f5b`). Evaluation scripts reconstruct complete routes, recompute their native-scale costs, and verify feasibility under the constraints of the evaluated formulation.

D Inference-Time Augmentation

Table A6 isolates the effect of inference-time augmentation on the frozen $1{,}000$-instance MDCVRP test set with $D{=}3$ and $N{=}100$. Both models use the complete $K = N + D - 1 = 102$ multi-start set. Without augmentation, GenRoute obtains a gap of 4.90%, compared with 4.55% for MD-MTA. Under their standard inference protocols, GenRoute uses eight dihedral transformations, whereas MD-MTA uses ten transformations: the eight dihedral transformations and two depot rotations.

Augmentation reduces the GenRoute gap by 2.64 percentage points and the MD-MTA gap by 1.91 percentage points. GenRoute has the lower objective under the standard augmented protocol, while MD-MTA is lower without augmentation.

Inference time. Table A7 reports wall-clock inference time per instance for the executable models, measured in one session on a single NVIDIA H200 NVL with no other process on the GPU, each model under its standard decoding protocol.

E Depot-Count and Problem-Size Generalization

Figure A2 visualizes the grid reported numerically in the main paper and additionally includes the $D = 3$-trained Stage-2 checkpoint. The comparison uses 200 matched MDCVRP instances per

Model	Protocol	Rollouts	s / instance
GenRoute (Stage 1)	1 instance per pass, augmentation 8	816	0.062
GenRoute-DM	1 instance per pass, augmentation 8	816	0.061
GenRoute (Stage 1)	10 instances per pass, augmentation 8	816	0.011
GenRoute (Stage 1)	50 instances per pass, augmentation 8	816	0.007
MD-MTA	1 instance per pass, 10 transformations	1,020	0.076
MD-MTA	10 instances per pass, 10 transformations	1,020	0.017
FiLMMeD	1 instance per pass, augmentation 8	816	0.505
FiLMMeD	10 instances per pass, augmentation 8	816	0.315
DeepMDV	CPU, its own evaluator, greedy, 10 per pass	–	0.365

Table A7: Wall-clock inference time per instance on the 1,000-instance MDCVRP pool (D=3, N=100), one NVIDIA H200 NVL, single session, no other process on the GPU. FiLMMeD's evaluator loads the grid data, so its row uses the D=3, N=100 cell (200 instances of the same distribution). DeepMDV's evaluator runs only on CPU in our environment; its row is its own reported serial duration per instance on 100 instances and is not comparable with the GPU rows.

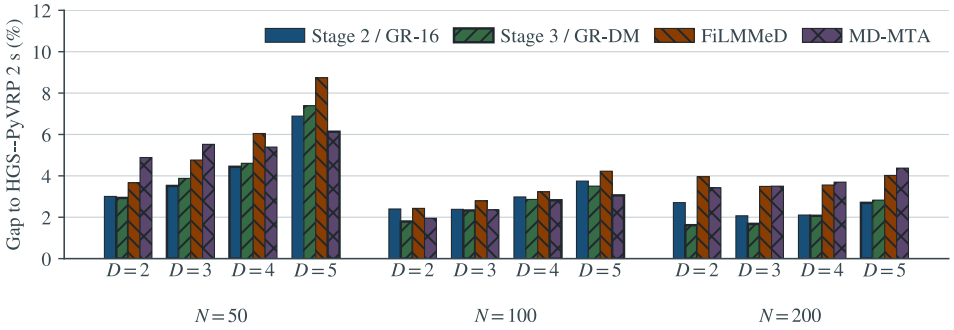


Figure A2: Depot-count and problem-size generalization on matched MDCVRP instances. Bars show mean gaps to 2 s HGS–PyVRP over 200 instances per setting; outlined bars mark the lowest gap. All models were trained at $N = 100$. Stage 3 was trained with $D \in \{1, 2, 3\}$, whereas Stage 2, FiLMMeD, and MD-MTA were trained at $D = 3$. Hatches and outlines make the systems distinguishable without color. Consequently, $D \in \{4, 5\}$ is a depot-count extrapolation for every model.

setting, the same 2 s HGS–PyVRP references, and strictly independent batch-size-one decoding for both refreshed checkpoints.

F Additional Multi-Variant Evaluation

This section reports the CVRPLIB table and two detailed comparisons that are summarized, but not fully tabulated, in the main paper: the paired 16-variant comparison with FiLMMeD and the mixed-backhaul half of RouteFinder's released single-depot suite. Results already shown in full in the main paper, including the multi-depot suite, are not repeated.

F.1 Evaluation Protocol

The primary 16-variant comparison uses a fresh suite generated after the model architecture, observation design, and evaluation protocol had been fixed. It contains 100 instances for each of the 16 variants. GenRoute-DM and FiLMMeD (Corrêa et al., 2026a) are evaluated on identical instances using unaugmented greedy decoding.

GenRoute-DM is evaluated one underlying instance at a time. Its batch-normalization statistics therefore depend only on representations belonging to that instance, so a prediction cannot depend on other test instances placed in the same evaluation batch.

For RouteFinder's released single-depot suite (Berto et al., 2025), we use the provided instances and bundled HGS–PyVRP references. Models use augmentation 8, greedy decoding, and their full multi-start budgets. The baseline values are obtained on the released test sets using the corresponding released checkpoints and native evaluation implementations.

Variant	GR-16^z	GR-DM^m	FiLM^z	MoS-C	MoS-R	RF-TE	RF-MoE	MVMoE	RF-POMO	MTPOMO
VRPMB	4.81	**2.14**	5.67	10.24	9.74	9.95	10.27	10.71	10.66	11.09
OVRPMB	6.48	**3.83**	11.92	19.08	18.80	18.91	19.45	20.40	20.31	20.64
VRPMBL	7.29	**3.14**	6.97	10.46	10.21	10.18	10.26	11.17	10.88	11.75
VRPMBTW	7.82	**4.52**	13.77	11.42	11.14	10.82	10.63	11.07	11.93	11.05
OVRPMBL	6.47	**3.84**	11.70	18.95	18.66	18.87	19.39	20.11	20.32	20.45
OVRPMBTW	6.35	**3.56**	12.80	8.93	8.90	8.76	8.78	9.35	9.75	9.10
VRPMBLTW	8.11	**5.00**	14.40	11.15	11.11	10.83	10.61	11.18	11.91	11.28
OVRPMBLTW	6.34	**3.57**	12.71	8.90	9.06	8.78	8.77	9.32	9.73	9.14
Mean	6.71	**3.70**	11.24	12.39	12.20	12.14	12.27	12.91	13.19	13.06

Table A8: Mean percentage gaps to the bundled 2 s HGS–PyVRP references on RouteFinder's eight released mixed-backhaul variants ($D = 1$, $N = 100$; 1,000 instances each). All methods use augmentation 8, greedy decoding, and their full multi-start budgets. ^{z}Training at $D = 3$ only; ^{m}training with $D \in \{1, 2, 3\}$. Boldface marks the lowest gap. Because training exposure differs, this is a coverage-and-transfer comparison.

Model	Training scope	Mean gap	Feas.
GenRoute-DM	16 variants, $D \in \{1, 2, 3\}$	$+5.5\%$	100%
MoSES	16 single-depot variants	**+4.4%**	100%
RF-TE	16 single-depot variants	$+4.6\%$	100%

Table A9: Transfer to 14 CVRPLIB X-set instances (Uchoa et al., 2017) with $100 \leq N \leq 200$, without benchmark-specific fine-tuning. We report mean gaps to the best-known solutions under the official integer EUC_2D metric. All methods use augmentation 8, greedy decoding, and their full multi-start budgets; all returned solutions are feasible.

F.2 CVRPLIB Transfer

Without benchmark-specific fine-tuning, GenRoute-DM obtains a mean gap of 5.5% across the 14 selected CVRPLIB instances (Table A9), with all solutions feasible. MoSES and RF-TE obtain lower gaps of 4.4% and 4.6%, respectively. GenRoute-DM therefore trails these single-depot-trained models by 1.1 and 0.9 percentage points while using one checkpoint across depot counts and all 16 variants.

F.3 Mixed-Backhaul Single-Depot Variants

Table A8 complements the eight single-depot variants reported in the main paper with RouteFinder's remaining eight mixed-backhaul variants. GenRoute-16 and FiLMMeD transfer zero-shot from mixed-backhaul training at $D = 3$, whereas GenRoute-DM subsequently samples $D \in \{1, 2, 3\}$. The released RouteFinder, MoSES, MVMoE, and MTPOMO checkpoints were not trained on this family (Berto et al., 2025; Pan et al., 2025; Zhou et al., 2024; Liu et al., 2024); the table therefore compares coverage and transfer rather than isolating architecture.

Mixed-depot training lowers GenRoute's mean gap from 6.71% to 3.70%, showing improved transfer across depot counts while retaining coverage of all 16 variants.

F.4 Paired Comparison Across Sixteen Variants

Table A10 gives the complete per-variant comparison between the final mixed-depot GenRoute-DM checkpoint and FiLMMeD. Across the 1,600 paired instances, GenRoute-DM obtains an aggregate mean objective of 13.7536, compared with 14.0743 for FiLMMeD, a reduction of 2.3%. GenRoute-DM has the lower mean objective in 9 of 16 variants and the lower per-instance cost in 1,083 of the 1,600 comparisons.

The improvement is concentrated on variants containing mixed backhauls, particularly when combined with open routes or time windows. FiLMMeD has the lower mean on seven variants, including a near tie on MDOVRPLTW; GenRoute-DM retains the lower aggregate mean without using a variant-specific checkpoint.

Variant	GR-DM	FiLM	GR wins	Variant	GR-DM	FiLM	GR wins
MDCVRP	12.6622	**12.6420**	48/100	MDVRPMBL	**11.3314**	11.4511	59/100
MDOVRP	8.4612	**8.4518**	48/100	MDVRPMBTW	**21.2855**	22.6305	97/100
MDVRPMB	**11.4345**	11.5582	67/100	MDVRPLTW	20.8924	**20.6966**	37/100
MDVRPL	12.4375	**12.4150**	47/100	MDOVRPMBL	**8.0874**	8.4162	96/100
MDVRPTW	21.0406	**20.8214**	36/100	MDOVRPMBTW	**13.8077**	14.7280	99/100
MDOVRPMB	**8.0386**	8.3872	97/100	MDOVRPLTW	13.7690	**13.7668**	51/100
MDOVRPL	**8.5157**	8.5486	59/100	MDVRPMBLTW	**20.9591**	22.3941	95/100
MDOVRPTW	13.7248	**13.7157**	48/100	MDOVRPMBLTW	**13.6103**	14.5657	99/100
Mean / total	**13.7536**	14.0743	1083/1600				

Table A10: Paired unaugmented-greedy results on the untouched suite (100 shared instances per variant). GR-DM is the final mixed-depot checkpoint, FiLM denotes FiLMMeD, and "GR wins" is the number of instances on which GR-DM has lower cost. Boldface marks the lower mean objective; MB denotes mixed backhauls.

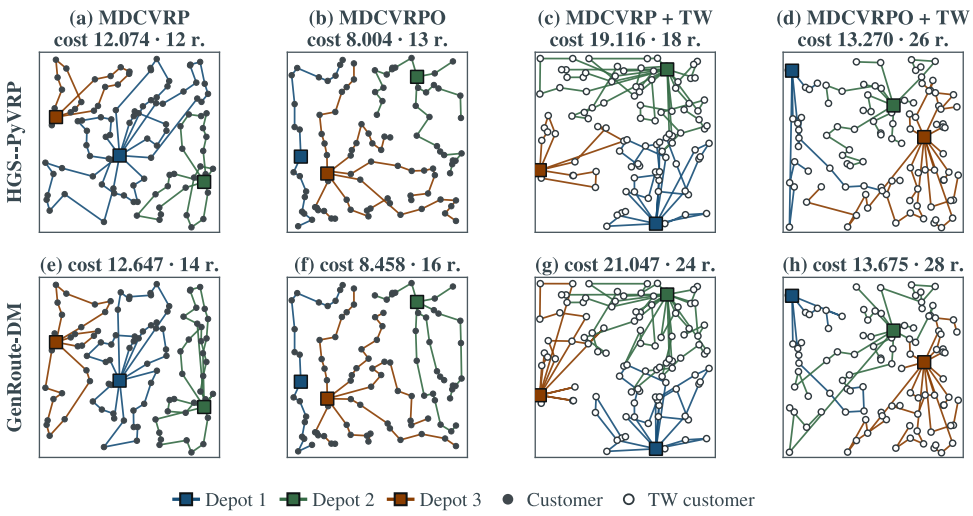


Figure A3: Matched route constructions on four frozen unseen $D = 3, N = 100$ instances across the O/TW attribute grid. Columns show MDCVRP, MDCVRPO (open only), MDCVRP with time windows, and MDCVRPO with time windows; rows compare 2 s HGS–PyVRP with the final GenRoute-DM checkpoint. The two panels in each column use the identical instance. Open-route polylines intentionally stop at the last customer rather than drawing a return edge. Each instance is selected solely as the GenRoute-DM objective closest to that variant's 100-instance mean, before running HGS, so selection is independent of the reference solver. Displayed costs are rescored from the retained routes in the original continuous coordinates; HGS is optimized under the paper's 10^4 integer-coordinate protocol with conservatively scaled time constraints. Squares denote depots, filled circles denote ordinary customers, hollow circles denote customers with time windows, and route color identifies the owning depot.