REAL-TIME NEURAL VEHICLE ROUTING UNDER DELIVERY COMMITMENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vehicle routing in the real world is inherently dynamic, where travel times fluctuate and orders are cancelled or released mid-operation, demanding real-time optimization. Existing Neural Vehicle Routing (NVR) methods can infer high-quality routes within milliseconds, yet they remain static and cannot respect delivery commitments under dynamic events. In this paper, we propose a Real-Time NVR (RT-NVR) approach, which extends NVR from static optimization to online decision-making. During training, a View-Comparative Policy Optimization (VCPO) method is designed to train an encoder-decoder model with state broadcasting, where the performance gap between geometrically equivalent views is turned into a learning signal via a contrast-adaptive baseline. In addition, a Spatiotemporal Compatibility-Guided Sampling (SCGS) strategy is devised to inject spatiotemporal priors into decoding, modulating logits toward promising actions. During real-time deployment, a Delivery Commitment Set-based Masking (DCSM) mechanism derives the vehicle load peak and maximum slack time of the delivery commitment set for a vehicle, establishing a provably safe decision boundary. Extensive experiments demonstrate that VCPO outperforms state-of-the-art training methods, and that the real-time approach reduces travel cost, fleet size, and time delay by 19.16%, 42.14%, and 96.73%, respectively, over heuristic methods. Statistical analyses further reveal that state broadcasting carries the most salient routing knowledge, and that the model parameters evolve along a low-dimensional manifold whose top 10 principal components capture 90.6% of the variance.

1 INTRODUCTION

The vehicle routing problem (VRP) is a fundamental combinatorial optimization problem (COP) that is pervasive in logistics, transportation, and supply chain management (Pillac et al., 2013). The classical VRP assumes a static environment and is solved in one go. However, real-world scenarios are inherently dynamic and uncertain: travel times fluctuate spatiotemporally, customer orders might be cancelled, and new orders might be released during operation (Psaraftis et al., 2016). This necessitates millisecond-level real-time optimization to maintain service quality and operational efficiency, giving rise to real-time VRP (RT-VRP) (Liang et al., 2024). Traditional re-optimization is typically triggered by time or events, and each real-time scenario snapshot can be viewed as a static problem. Although static optimization algorithms, such as exact and heuristic methods, can be used to solve the problem, they rely on time-consuming iterative search and are unable to meet millisecond-level response requirements (Vidal et al., 2013; Helsgaun, 2017).

Based on Neural Combinatorial Optimization (NCO) (Bengio et al., 2021), Neural Vehicle Routing (NVR) uses a learnable routing policy to replace the time-consuming optimization process. The Pointer Network (Ptr-Net) (Vinyals et al., 2015) first introduced a pointer mechanism into a sequence-to-sequence architecture to solve COPs. Building on this, the Attention Model (AM) (Kool et al., 2019) established the encoder-decoder as the dominant architecture. Policy Optimization with Multiple Optima (POMO) (Kwon et al., 2020) improved training by generating diverse trajectories starting from different nodes, and subsequent works such as light encoder and heavy decoder (LEHD) (Luo et al., 2023) and RouteFinder (Berto et al., 2025) extended toward cross-scale and cross-task generalization, respectively. These NCO methods encode absolute customer locations, yet the optimal solution is invariant under translation, rotation, or reflection. Replacing absolute coordinates with relative distances restores this invariance but discards useful spatial inductive biases, and is thus rarely

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.

adopted. Some recent work resorts to training-time data augmentation (Fang et al., 2024), symmetry-aware training (Kim et al., 2022), or test-time inference augmentation (Bdeir et al., 2022), yet little effort has focused on learning better policies from the performance gap between geometrically equivalent views. More importantly, existing NVR approaches still focus on solving static VRPs, and a method for real-time scenarios remains scarce (Chung et al., 2025; Wu et al., 2024). When a dynamic event occurs mid-route, each vehicle is at a different location and carries committed delivery goods that cannot be reassigned to other vehicles during operation. The problem in each real-time scenario snapshot is therefore reformulated as a VRP with fixed start nodes and delivery commitments (VRP-FSDC), a setting that has received little attention in the NVR literature.

In this paper, we propose a real-time neural vehicle routing (RT-NVR) approach. An encoder-decoder model with state broadcasting is trained offline and deployed online to make real-time decisions under safe decision boundaries. The core contributions are as follows:

- **View-Comparative Policy Optimization (VCPO).** Geometrically equivalent views are generated for each training instance. The performance gap between these views is then exploited as a training signal to assign a contrast-adaptive baseline, encouraging policy exploration on better views while penalizing the worse ones.
- **Spatiotemporal Compatibility-Guided Sampling (SCGS).** A spatiotemporal compatibility matrix is computed to inject spatial and temporal priors into the decoding process, guiding the model to sample actions that are conducive to generating high-quality trajectories.
- **Delivery Commitment Set-based Masking (DCSM).** A delivery commitment set (DCS) is constructed for each vehicle, and its peak load and maximum slack time are theoretically derived to mask infeasible candidates, providing a safe boundary for model decision.

2 Related Work

Static Optimization Methods. VRP is extensively studied as a static optimization problem, where all information is known in advance and remains unchanged during operation (Archetti et al., 2026). Exact algorithms are limited to small-scale instances (Praxedes et al., 2024). Heuristic algorithms reduce the computational complexity by sacrificing optimality (Friedrich & Elbert, 2022). For example, LKH-3 (Helsgaun, 2017) is a leading heuristic solver with variable-depth local search. Meta-heuristic algorithms explore the search space via evolution strategies, and combining evolutionary algorithms (EAs) with neighborhood search is currently a mainstream strategy. Such advanced algorithms include Hybrid Genetic Search with Adaptive Diversity Control (HGSADC) (Vidal et al., 2013) and its extension with the SWAP* neighborhood (Vidal, 2022). Recent advances also include Estimation of Distribution Algorithms (EDAs) for electric vehicle routing (Shen et al., 2021), and green vehicle routing (Wang et al., 2026).

Real-Time Optimization Methods. In real-world scenarios, random events may dynamically emerge during operation. Time-driven methods re-optimize the current route based on the latest information observed at fixed intervals, while event-driven methods are triggered once an event occurs (Pillac et al., 2013). In practice, major logistics enterprises like Cainiao and Meituan (Chen et al., 2024; Liang et al., 2024) resort to lightweight time-driven online adjustments. In addition, rolling horizon methods incorporate a look-ahead window to predict stochastic information (Bjelić et al., 2022; Li et al., 2024), and stochastic programming methods model uncertainty through probability distributions (Alvarez et al., 2024; Ghosal et al., 2024; Ota & Fukasawa, 2025). However, customer-side changes exhibit strong randomness that is difficult to reliably predict or model.

Neural Vehicle Routing Methods. The Pointer Network (Ptr-Net) (Vinyals et al., 2015) introduced a pointer mechanism for selecting elements from an input set. Bello et al. (2017) replaced supervised learning with reinforcement learning to eliminate the need for optimal-solution labels, and Kool et al. (2019) established the encoder-decoder as a dominant backbone. POMO (Kwon et al., 2020) further improved training by sampling trajectories from different fixed start nodes. For large-scale routing, Li et al. (2021) trained a model to decompose large-scale instances. Luo et al. (2023) employed a heavy decoder, while Fang et al. (2024) aggregated nested views of multiple neighborhood scales, improving cross-scale generalization. Luo et al. (2025) further introduced self-improved training to generate pseudo-labels via local reconstruction, enabling large-scale training. For cross-task

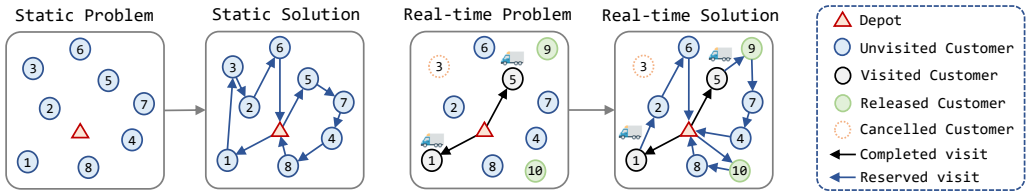


Figure 1: Comparison between static optimization and real-time optimization. Static optimization produces a one-shot initial solution. Real-time optimization replans each vehicle's route while respecting the constraints of fixed start nodes and delivery commitments when new events emerge.

generalization, Berto et al. (2025) proposed RouteFinder to solve multi-attribute VRPs, and Yu et al. (2026) proposed CoEKS with a knowledge-sharing mechanism. In terms of geometric invariance, Bdeir et al. (2022) enhanced inference via isometric transformations, and Zhang et al. (2024) re-scaled attention weights by relative distances. In addition, through probing, Zhang et al. (2025) empirically verified the capability of NVR models to perceive spatial distances and temporal tightness. Despite these advances, they mostly target static VRPs. Even though RL-based dynamic methods (Joe & Lau, 2020; Ma et al., 2021) can periodically re-optimize routes and re-assign orders, they ignore the delivery commitment constraint, leaving an open gap for an NVR method that handles real-time events while respecting delivery commitments, especially for simultaneous pickup and delivery.

3 Preliminaries

3.1 Vehicle Routing Problem with Pickup, Delivery, and Time Windows

In this study, we consider a generic VRP variant, i.e., VRP with pickup, delivery, and time windows (VRPPDTW). In this problem, a fleet of vehicles, each with a capacity of Q, is employed to serve N customers while respecting operational constraints. The maximum size of the fleet is denoted by $K_{\max}$. Each customer i has a location $\mathbf{c}_i = [c_{i,1}, c_{i,2}]$, a delivery demand d_i, a pickup demand p_i, a time window $[e_i, l_i]$, and a service time s_i. A solution is represented as $S = \{0, R_1, 0, R_2, 0, \ldots, 0, R_K, 0\}$, where 0 represents the depot, K is the number of vehicles assigned, and $R_k = \{r_{k,1}, r_{k,2}, \ldots, r_{k,I_k}\}$ denotes the route of vehicle k, $r_{k,0} = r_{k,I_k+1} = 0$, with $r_{k,i}$ being the index of the i-th customer visited and I_k being the number of customers assigned.

Therefore, the problem is formulated as:

$$\min \sum_{k=1}^{K} \sum_{i=1}^{I_k+1} \left\| \mathbf{c}_{r_{k,i-1}} - \mathbf{c}_{r_{k,i}} \right\|_2 \tag{1}$$

$$\text{s.t. } K \leq K_{\max} \tag{2}$$

$$\sum_{j=1}^{I_k} d_{r_{k,j}} + \sum_{j=0}^{i} (p_{r_{k,j}} - d_{r_{k,j}}) \leq Q, \quad \forall k = 1, \ldots, K, \quad \forall i = 0, \ldots, I_k \tag{3}$$

$$a_{r_{k,i}} \leq l_{r_{k,i}}, \quad \forall k = 1, \ldots, K, \quad \forall i = 1, \ldots, I_k \tag{4}$$

$$\sum_{k=1}^{K} \sum_{i=1}^{I_k} \mathbb{1}_n(r_{k,i}) = 1, \quad \forall n = 1, \ldots, N \tag{5}$$

where Equation (1) is the optimization objective of minimizing travel cost; Equation (2) is the vehicle number constraint; Equation (3) is the vehicle capacity constraint; Equation (4) is the time window constraint, where $a_{r_{k,i}}$ is the arrival time at customer $r_{k,i}$; Equation (5) is the single-visit constraint, where $\mathbb{1}_n(r_{k,i})$ is an indicator function that equals 1 if customer n is the i-th customer served by vehicle k, and 0 otherwise. Based on the above definition, real-time optimization has to solve a VRPPDTW with fixed start nodes and delivery commitments upon detecting new events, which is denoted as VRP-FSDC for short. As illustrated in Figure 1, real-time decisions must respect their current locations since vehicles are en route, and each vehicle must complete the delivery tasks that were committed before departure.

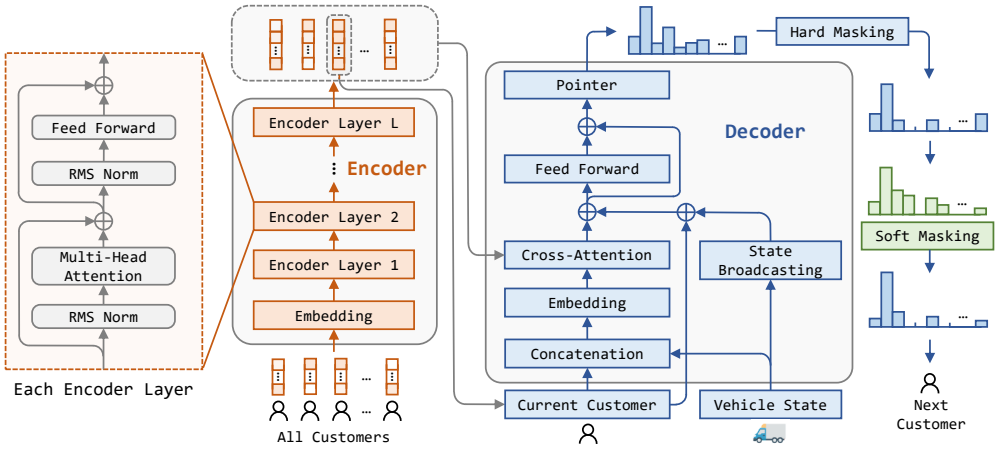


Figure 2: Overview of the encoder-decoder architecture with state broadcasting.

4 METHODOLOGY

4.1 MODEL ARCHITECTURE

An overview of model architecture is shown in Figure 2. The encoder and decoder are briefly described below; more details are provided in Appendix A.1.

Encoder. The encoder performs global feature extraction. It takes customer features as input with the depot being a special customer. The feature vector of customer i is $[c_{i,1}, c_{i,2}, d_i, p_i, e_i, l_i, s_i]$. A linear projection first maps each vector to a d_{emb}-dimensional embedding space. Each embedding is then passed through L layers, each consisting of a multi-head self-attention sub-layer and a feed-forward sub-layer, both with residual connections. Root mean square normalization (RMSNorm) is applied before each sub-layer to enhance network stability. The output of the encoder is a set of customer embeddings $\{\mathbf{e}_i \in \mathbb{R}^{d_{\text{emb}}}\}$ that capture the complex interactions among customers.

Decoder. The decoder is invoked once per step to decide the next visit for a vehicle. The current vehicle state is represented as $[t, \bar{Q}_k^{\text{p}}, \bar{Q}_k^{\text{d}}]$, where t is the current time, $\bar{Q}_k^{\text{p}}$ is the residual pickup capacity, and $\bar{Q}_k^{\text{d}}$ is the residual delivery capacity. The decoder first concatenates the current embedding $\mathbf{e}_u$ with the current vehicle state and projects the concatenated vector to a d_{emb}-dimensional embedding space. A cross-attention mechanism is then employed over the embeddings of committed yet unvisited customers, active pickup customers and the depot, producing a context vector. A residual connection broadcasts the vehicle state into the context, and the result is passed through a feed-forward layer to obtain a query vector. Finally, the query vector is combined with the customer embeddings via a dot product to obtain a logit vector $\boldsymbol{\pi}$. The logits are then modified by masking, followed by a softmax to produce a probability distribution, from which the next customer is sampled.

Hard Masking. A load margin ρ_k is maintained for each vehicle k, initialized as Q and updated after each reservation by $\rho_k \leftarrow \min\big(\rho_k - d_i,\ Q - P_k - p_i\big)$, where d_i and p_i are the delivery and pickup demands of the just-reserved customer i, respectively, and P_k is the cumulative pickup demand before reserving customer i. Therefore, an unvisited candidate j is feasible if $d_j \le \rho_k, p_j \le Q - P_k$, and $t + \tau_{i,j} \le l_j$, where $\tau_{i,j}$ is the travel time from customer i to customer j. The load margin reduces the computational complexity of the capacity feasibility checking from $O(N)$ to $O(1)$ per step. The theoretical derivation is provided in Appendix A.2. Actions selecting infeasible customers are masked, i.e., assigned a logit of $-\infty$, and the probability vector after softmax is $\bar{\boldsymbol{\pi}}$.

Soft Masking. Spatiotemporal Compatibility-Guided Sampling (SCGS) further improves rollout quality. Specifically, a compatibility matrix $\mathbf{C} = \phi(\mathbf{H}^{\text{S}}) \odot \phi(\mathbf{H}^{\text{T}})$ is precomputed, where $\mathbf{H}^{\text{S}}$ and $\mathbf{H}^{\text{T}}$ are the spatial and temporal compatibility matrices, respectively, $\phi(\cdot)$ denotes row-wise normalization, and $\odot$ is the Hadamard product. The matrix $\mathbf{H}^{\text{S}}$ measures the spatial compatibility between customers, with each entry defined as:

$$H^{\text{S}}_{i,j} = \lambda^{o_{i,j}} \cdot \big(\max_{k \neq i}(D_{i,k}) - D_{i,j}\big), \tag{6}$$

where $D_{i,j} = \|\mathbf{c}_i - \mathbf{c}_j\|_2$ is the Euclidean distance between customers i and j, $o_{i,j}$ is the rank of customer j from customer i by ascending distance, starting from 0, and $\lambda \in (0, 1)$ is a discrimination factor. The matrix $\mathbf{H}^{\mathrm{T}}$ measures the temporal compatibility between customers. For a customer pair (i, j), if $e_i + s_i + D_{i,j}/v_{\max} > l_j$, $H^{\mathrm{T}}_{i,j} = 0.0$, where $v_{\max}$ is the maximum speed; else if $l_i + s_i + D_{i,j}/v_{\min} \geq e_j$, $H^{\mathrm{T}}_{i,j} = 1.0$, where $v_{\min}$ is the minimum speed, for simplicity, $v_{\max}$ = $v_{\min}$ = 1.0; otherwise, it is computed as $H^{\mathrm{T}}_{i,j} = 1/(1 + \delta_{i,j})$, where $\delta_{i,j}$ is a time gap:

$$\delta_{i,j} = e_j - (l_i + s_i + D_{i,j}/v_{\min}). \tag{7}$$

Finally, the probability vector is further modulated as:

$$\tilde{\boldsymbol{\pi}} = \phi\big(\boldsymbol{\pi} \odot \mathbf{C}_{u,:}\big). \tag{8}$$

4.2 Offline Training

The model is trained with a View-Comparative Policy Optimization (VCPO) method. A trajectory, i.e., a sequence of visits, is sampled from the policy model and its advantage is computed based on a Contrast-Adaptive Baseline (CAB) derived directly from rewards, without training a critic model.

Equivalent View Comparison. To turn the geometric invariance from a passive prior into an active optimization driver, we propose VCPO. The idea is to generate equivalent views for each training instance and exploit the performance gap between them as a training signal. For each training instance, an equivalent view is randomly generated by applying one of the eight rigid transformations to coordinates; these transformations form the dihedral group, i.e., reflections and rotations by multiples of $90°$. Trajectories are then independently sampled from the original view and the transformed view, yielding two groups: $\{\xi^+_g\}^G_{g=1}$ of the better view with higher mean reward μ^+ and $\{\xi^-_g\}^G_{g=1}$ of the worse view with lower mean reward μ^-, where ξ^+_g and ξ^-_g are the g-th trajectories from the better and worse views, respectively, and the reward is defined as the negative travel cost.

Contrast-Adaptive Baseline. The better view receives an unbiased baseline equal to its own mean, encouraging continual policy exploration; the worse view receives a biased baseline pulled toward the better view's mean, penalizing the over-reliance on absolute coordinates. The corresponding loss is:

$$\mathcal{L}_{\text{VCPO}} = -\frac{1}{G}\sum_{g=1}^{G}(J^+_g - \eta^+)\log p(\xi^+_g) - \frac{1}{G}\sum_{g=1}^{G}(J^-_g - \eta^-)\log p(\xi^-_g), \tag{9}$$

where $p(\xi)$ denotes the probability of trajectory ξ under the policy, J^+_g and J^-_g are the rewards of the g-th trajectories from the better and worse views, respectively, $\eta^+ = \mu^+$, $\eta^- = \gamma\mu^+ + (1-\gamma)\mu^-$, with $\gamma = 0.5 + 0.5\tanh(\kappa\Delta)$, and $\Delta = \mu^+ - \mu^- \geq 0$ is the performance gap, with $\kappa > 0$ controlling the sensitivity. As $\Delta \to \infty$, $\gamma \to 1$, exerting strong contrastive pressure on the worse view; as $\Delta \to 0$, $\gamma \to 0.5$. This adaptive mechanism thus balances low gradient variance with high signal-to-noise ratio, with a theoretical analysis provided in Appendix A.3.

4.3 Online Deployment

The online deployment consists of two levels: an execution level monitors dynamic events and executes the current routing schedule, while a planning level re-optimizes the routes in response to dynamic events. Before real-time operation, a static routing schedule is obtained in advance via PyVRP solver (Vidal, 2022). Accordingly, a Delivery Commitment Set (DCS) $B_k = \{b_1, b_2, \ldots, b_M\}$ is extracted from R_k by removing all pure pickup customers, where M is the size of the DCS. It records the unvisited customers to which this vehicle is committed. The feasibility of R_k is a sufficient condition for the feasibility of B_k (Appendix A.4.1). Under the constraints of DCS, the trained NVR model replans routes in an event-driven manner.

Execution Level. The execution level advances in discrete time steps. At each vehicle arrival, the execution level updates real-time states. It continuously monitors three types of dynamic events: 1) a pure pickup order is released and is added to the order pool; 2) an existing order is cancelled and is removed from the order pool; 3) a vehicle arrives at a customer i with arrival time $a_i > l_i$. When no dynamic events are detected, all vehicles follow their current routes without interruption. Upon detecting any of these events, the execution level triggers a real-time routing of the planning level.

Planning Level. According to the latest operational information and vehicle state, the planning level invokes the trained model to solve a VRP-FSDC. At each step, in order to respect real-time constraints, a Delivery Commitment Set-based Masking (DCSM) strategy is designed to construct a candidate set $\mathcal{C}_k$ for the next visit based on two aspects:

1. **Vehicle Load Peak.** A feasible pickup customer j must satisfy $p_j + \max\{\ell, \ell_1, \ldots, \ell_M\} \leq Q$, where ℓ is the current load and ℓ_i is the load at the i-th customer in the DCS, ensuring that the vehicle load peak after insertion does not exceed the vehicle capacity. The theoretical foundation is provided in Appendix A.4.2.
2. **Maximum Slack Time.** To avoid delay propagation in committed deliveries, a maximum slack time along the DCS of the vehicle is calculated as $\Theta = \min_{2 \leq i \leq M} \theta_i$, where

$$\theta_i = l_{b_i} - \sum_{z=2}^{i-1} s_{b_z} - \sum_{z=2}^{i} \tau_{b_{z-1}, b_z}. \tag{10}$$

The latest feasible arrival time at b_1 is then $l_{\max} = \min\{l_{b_1}, \Theta - s_{b_1}\}$. Accordingly, a feasible pickup customer j must satisfy $t + \tau_{u,j} \leq l_j$ and $\max\{t + \tau_{u,j}, e_j\} + s_j + \tau_{j,b_1} \leq l_{\max}$, ensuring remaining customers in B_k can be visited within their time windows. The theoretical foundation is provided in Appendix A.4.3.

All feasible pickup customers form $\mathcal{P}_k$, and the final candidate set is $\mathcal{C}_k = \{b_1\} \cup \mathcal{P}_k$. By including b_1, the model can flexibly decide whether to insert a feasible pickup or to visit the next delivery, avoiding inefficient detours. At each decoding step, the current customer attends to customers in B_k, active pickup customers and the depot via cross-attention. The mask restricts the output to only $\mathcal{C}_k$, ensuring safe decisions. When the DCS is exhausted and no more pickup can be feasibly visited, the vehicle returns to the depot. The planning level can naturally determine whether new vehicles need to be dispatched from the depot at each decision cycle. After all active customers have been routed, the planning level returns the updated routes to the execution level for step-by-step execution.

5 Experiments

5.1 Setup

Real-Time Instance Generation. Following the pipeline in SVRPBench (Heakl et al., 2025), a multi-center Gaussian mixture distribution that mimics urban demand hotspots is used to generate problem instances, with the complete procedure provided in Appendix B.1. Given a static problem instance, a stream of dynamic events is simulated: each pickup order has a fixed probability f_c of being cancelled before service, $f_i N$ new pickup orders are dynamically released during execution, and travel time $\tau_{ij} = \max\{D_{ij}, \mathcal{N}(2D_{ij}, 0.2D_{ij})\}$. Details are available in Appendix B.2.

Comparative Training Methods. (1) REINFORCE (Bello et al., 2017), denoted as REIN for short; (2) POMO (Kwon et al., 2020); (3) ITDA (Fang et al., 2024), an isometric transformation-based data augmentation method; and (4) ITAB, a variant of ITDA using the view average as a shared baseline.

Comparative Real-Time Algorithms. (1) Saving Value-based Insertion (SVI) (Anuar et al., 2022), inserting a new customer with maximum distance savings; (2) an extension of SVI (SVI-II), reinserting all pure pickup customers; (3) DCSM-based Nearest Neighboring (DCSM-NN) (Demirbilek, 2021), selecting the nearest feasible customer as the next visit; (4) DCSM-based Iterative Local Search (DCSM-ILS) (Máximo & Nascimento, 2021), iteratively adjusting routes via local search operators.

Experiment configurations. The model consists of 6 encoder layers, 8 attention heads, a hidden dimension of 128, and a feed-forward dimension of 512. During training, each epoch generates 10,000 training instances on the fly, with 1,000 fixed validation instances drawn using a separate random seed. The batch size is set to 32, the numbers of epochs and trajectories are set to 300 and 50, respectively. The Adam optimizer is used with a learning rate of 3×10^{-4}. The discrimination factor λ is set to 0.9, and the sensitivity factor κ is set to 1.0. These hyperparameters are not exhaustively tuned because the study focuses on promoting NVR to real-time scenarios rather than ultimate performance improvement. Offline training is conducted on a single RTX 5090 GPU, and online deployment is executed on a laptop with an Intel Core CPU @ 2.1 GHz and 16 GB RAM.

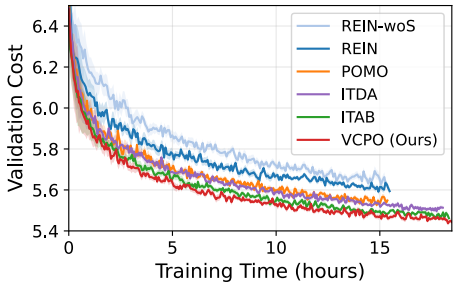


Figure 3: Validation cost convergence curves.

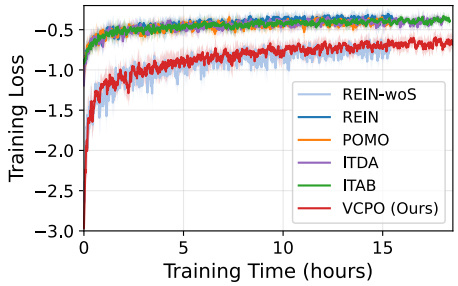


Figure 4: Loss variation curves.

Table 1: Average travel cost of the models trained by the five training methods on 100 random instances under different event frequency combinations (Best result per row is bolded).

f_c	f_i	VCPO (Ours)	ITAB	ITDA	POMO	REIN
0.1	0.1	**6.5362 ± 1.2469**	6.6386 ± 1.0395	6.6574 ± 1.2481	6.8843 ± 1.1087	6.6630 ± 1.2062
	0.2	**7.5226 ± 1.5514**	7.7198 ± 1.3379	7.8904 ± 1.4078	7.7645 ± 1.2987	7.7591 ± 1.3605
	0.3	**8.6101 ± 1.6235**	8.7952 ± 1.6930	8.6404 ± 1.8924	8.7161 ± 1.6804	8.7061 ± 1.6972
	0.4	9.4623 ± 2.0935	9.6623 ± 2.0222	**9.4259 ± 2.0812**	9.5688 ± 2.0056	9.5950 ± 1.8627
	0.5	10.6102 ± 2.4567	10.5437 ± 2.4926	10.5413 ± 2.4394	**10.3531 ± 2.7739**	10.6592 ± 2.2817
0.2	0.1	**6.2828 ± 1.0471**	6.3262 ± 1.1376	6.4092 ± 1.1249	6.5515 ± 1.1788	6.4162 ± 1.0616
	0.2	**7.2434 ± 1.2031**	7.4423 ± 1.2386	7.4221 ± 1.2871	7.3743 ± 1.2307	7.4417 ± 1.0993
	0.3	**8.2169 ± 1.6404**	8.2368 ± 1.3090	8.2532 ± 1.4533	8.2873 ± 1.4859	8.4297 ± 1.5280
	0.4	9.1889 ± 1.9239	9.0554 ± 1.7287	**8.9053 ± 1.4911**	9.1185 ± 1.7984	9.2697 ± 1.6284
	0.5	10.0149 ± 2.1184	**9.8136 ± 2.0265**	9.9381 ± 2.0421	9.9505 ± 2.2067	10.1682 ± 2.2886
0.3	0.1	6.0200 ± 1.0889	**6.0062 ± 0.9432**	6.1168 ± 0.9858	6.3158 ± 0.9439	6.0825 ± 0.9421
	0.2	**6.9727 ± 1.2386**	7.0859 ± 1.0502	7.1436 ± 1.1861	7.2111 ± 1.1234	7.0602 ± 1.1584
	0.3	**7.8319 ± 1.3564**	7.9467 ± 1.2458	8.2279 ± 1.6202	8.0198 ± 1.3001	8.0494 ± 1.4662
	0.4	8.7056 ± 1.6158	8.6654 ± 1.4498	**8.5764 ± 1.5261**	8.8833 ± 1.6587	8.8852 ± 1.7565
	0.5	9.5266 ± 1.8352	9.7804 ± 2.0902	**9.5048 ± 1.9977**	9.7946 ± 2.0089	9.8681 ± 2.0648
0.4	0.1	**5.7967 ± 0.9726**	5.8554 ± 0.9752	6.0523 ± 1.0646	6.0289 ± 1.0321	5.8708 ± 0.9438
	0.2	**6.6860 ± 1.0318**	6.8859 ± 1.0811	6.8614 ± 1.0373	7.0069 ± 1.0554	6.7907 ± 1.1819
	0.3	**7.4877 ± 1.3727**	7.6287 ± 1.1982	7.6671 ± 1.1886	7.6991 ± 1.2638	7.6607 ± 1.4675
	0.4	8.3905 ± 1.6897	8.4448 ± 1.3135	**8.3818 ± 1.4005**	8.4816 ± 1.2733	8.5564 ± 1.9211
	0.5	9.1441 ± 1.9857	**9.0530 ± 1.5916**	9.1022 ± 1.7816	9.1260 ± 1.4893	9.4175 ± 2.2808
0.5	0.1	**5.4145 ± 0.8836**	5.5334 ± 0.9241	5.5987 ± 0.9580	5.7317 ± 0.9929	5.5817 ± 0.9028
	0.2	**6.3398 ± 0.8804**	6.3902 ± 0.8998	6.5326 ± 1.0489	6.6431 ± 0.9736	6.4900 ± 1.0222
	0.3	7.1277 ± 1.1554	**7.0305 ± 1.0253**	7.3900 ± 1.2179	7.2845 ± 1.2756	7.3342 ± 1.3323
	0.4	**7.7679 ± 1.3548**	7.8590 ± 1.3442	8.0876 ± 1.3534	8.1504 ± 1.3700	8.0526 ± 1.4150
	0.5	**8.4639 ± 1.7465**	8.6551 ± 1.5432	8.7494 ± 1.7603	8.7887 ± 1.5453	9.0116 ± 1.8616

5.2 Main Results

Training performance comparison. The key components of the proposed VCPO are SCGS and CAB, where SCGS acts as a flexible plugin adopted by all methods; we therefore omit SCGS from the abbreviations and mark only its absence. The validation cost convergence during training is shown in Figure 3, where REIN-woS denotes a variant of REIN without SCGS. It can be seen that SCGS accelerates convergence by guiding the learning process with a domain prior, as REIN converges faster than REIN-woS. POMO improves upon REIN, achieving lower validation cost thanks to trajectory diversity. Due to the isometric transformation, ITDA, ITAB, and VCPO construct twice as many trajectories per batch. This increases the training time by only about $1/6$ with the assistance of parallel computation. Compared to ITDA and ITAB, VCPO achieves lower validation cost by using CAB to encourage exploration on better views while penalizing worse ones.

Training loss analysis. Loss variations are shown in Figure 4. Loss values stay negative because policies tend to assign high probability to decisions with positive advantages; their trend thus reflects learning behavior rather than performance. Comparative training methods with SCGS have a higher loss than REIN-woS. The reason is that SCGS raises the sampling probability of high-compatibility customers, bringing within-group trajectories closer and thus reducing the deviation of advantages. On this basis, VCPO introduces CAB to further reduce the negative advantage values from worse views, thus pulling the loss back to the REIN-woS level.

Table 2: Average performance of different real-time algorithms (Best result per row is bolded).

Metric	DCSM-NVR (Ours)	DCSM-ILS	DCSM-NN	SVI-II	SVI
Cost	$\mathbf{7.5725 \pm 1.7914}$	9.3668 ± 2.7914	9.6166 ± 2.8377	13.8719 ± 4.5369	12.4876 ± 5.2694
VN	$\mathbf{4.1744 \pm 1.3626}$	7.2450 ± 2.8146	7.2144 ± 2.8271	7.3931 ± 4.2462	5.8694 ± 3.4994
Delay	0.3457 ± 0.2588	0.3833 ± 0.3436	$\mathbf{0.3060 \pm 0.1603}$	18.7828 ± 9.0758	10.5950 ± 8.5231
Time (s)	0.0740 ± 0.0080	0.0927 ± 0.0123	$\mathbf{0.0129 \pm 0.0022}$	0.0659 ± 0.0261	0.0130 ± 0.0065

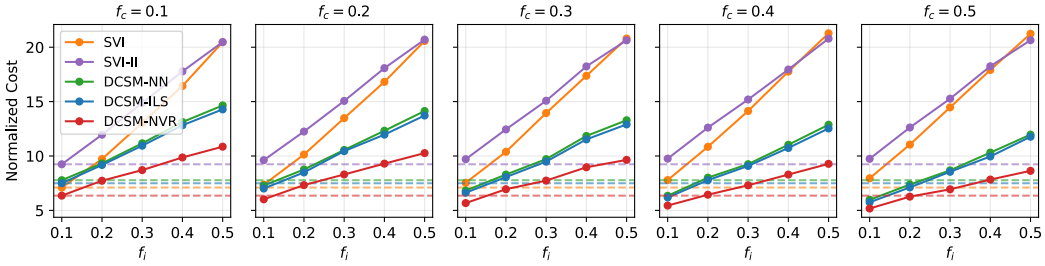


Figure 5: Average cost of different real-time algorithms under different event frequency combinations.

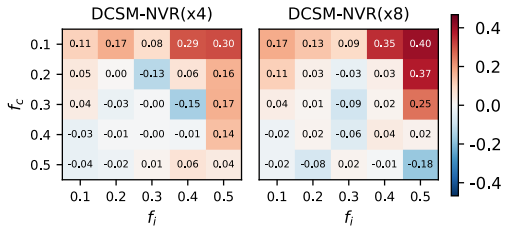


Figure 6: Performance gap of DCSM-NVR(x4) and DCSM-NVR(x8) relative to DCSM-NVR.

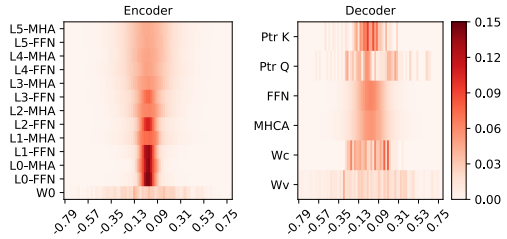


Figure 7: Per-module weight distribution of the encoder-decoder model with state broadcasting.

Real-time performance comparison with NVR methods. Table 1 reports the average performance of the model trained by different methods across all 25 (f_i, f_c) combinations. All methods attain lower cost as f_c grows and higher cost as f_i grows. The reason is that cancellations reduce the served workload and insertions increase the detouring risk. VCPO wins 13 of 15 combinations with $f_i \leq 0.3$, but only 2 of 10 with $f_i \geq 0.4$. ITDA and ITAB win the most remaining combinations, while POMO wins once, concentrated at high f_i. The underlying reason is that greedily replanning may restrict future decision flexibility.

Real-time performance comparison with heuristic methods. Table 2 lists the average real-time operating performance of different real-time algorithms, where VN indicates the number of vehicles used, and Figure 5 shows the details of travel cost, where the proposed RT-NVR approach is denoted as DCSM-NVR. It can be seen that SVI-II and SVI have higher cost and delay due to short-sighted local insertion and the absence of DCSM. DCSM-ILS shows slightly better performance than DCSM-NN. DCSM-NVR reduces the routing cost by 19.16% (vs. DCSM-ILS) and the vehicle number by 42.14% (vs. DCSM-NN), while the DCSM mechanism reduces the time delay by 96.73% (vs. SVI). In addition, as shown in Figure 5, cost increases with f_c for SVI-II and SVI, but decreases for other methods. This demonstrates that order cancellations make greedy insertion ineffective. DCSM-NVR is less sensitive to event frequency changes, which is attributed to the fact that DCSM provides a safe decision boundary while the NVR model makes far-sighted global planning under the boundary.

Inference augmentation analysis. DCSM-NVR(x4) and DCSM-NVR(x8) employ 4 and 8 inference-time isometric transformations to select the best trajectory and are compared against DCSM-NVR. The experimental result is summarized in Figure 6. Inference augmentation can slightly reduce cost under low f_i and high f_c. However, the performance gradually deteriorates as f_i increases or f_c decreases, which aligns with the findings in the previous experiment. The reason is that dynamic disturbances can disrupt the global structure assumed during inference. Slightly sacrificing the static improvement while leaving room for future adjustments may be a more promising strategy.

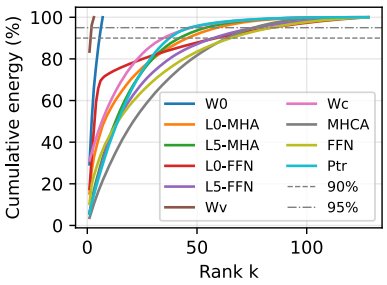


Figure 8: Cumulative energy curves.

Table 3: Module weight statistics across layers.

Module	std	< 0.01	< 0.05	> 0.5
Pointer	0.0991	7.99%	38.07%	0.00%
MCHA+FFN	0.1086	8.36%	39.33%	0.10%
Wc	0.1144	9.16%	42.60%	0.46%
Wv	**0.4774**↑	**2.08%**↓	**8.59%**↓	**23.44%**↑
Layer 5	0.1287	6.96%	33.86%	0.20%
Layer 4	0.1242	7.22%	34.86%	0.15%
Layer 3	0.1145	8.53%	39.63%	0.15%
Layer 2	0.0967	10.86%	49.20%	0.15%
Layer 1	0.0825	12.55%	56.38%	0.14%
Layer 0	**0.0713**↓	**14.03%**↑	**62.38%**↑	**0.13%**↓
W0	0.2639	2.64%	14.45%	4.00%

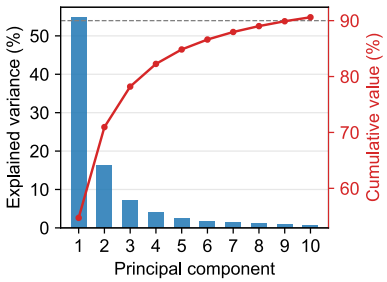


Figure 9: Explained variance of principal components of the evolution trajectory.

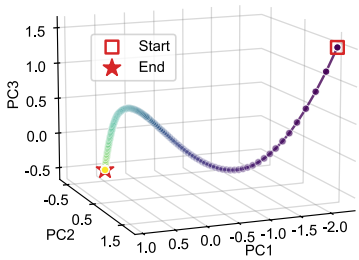


Figure 10: Evolution trajectory of model parameters in PC1–PC2–PC3 space.

Model parameter statistics. The model parameter statistics across layers are summarized in Figure 7 and Table 3. First, the state broadcasting (Wv) is the most active component, confirming that it carries decisive routing information. Encoder depth drives parameter magnitude, evidenced by the fact that std grows monotonically from 0.0713 (layer 0) to 0.1287 (layer 5) while the near-zero fraction drops from 14.03% to 6.96%. This demonstrates that deep layers concentrate stronger routing knowledge. Figure 8 visualizes the cumulative energy curves of key layers. It reveals that the weight matrices do not exhibit pronounced low-rank structure; the 95% threshold commonly falls in the range of rank 40–80, indicating rich internal structure in each weight matrix.

Evolution trajectory analysis. We collect 150 checkpoints with 1,437,568 parameters each during training and perform principal component analysis (PCA) on the flattened parameter vectors. The results are shown in Figure 9, indicating that PC1 alone explains 54.7% of the variance and the top 10 components cumulatively capture 90.6%. Figure 10 visualizes the evolution trajectory of the model parameters in PC1–PC2–PC3 space, revealing that the model evolves along a low-dimensional manifold in the parameter space; per-module PCA is available in Appendix C.3. In summary, *what is finally learned by the model is complex, but how it is learned follows a simple path*.

6 Conclusion

In this paper, we proposed an RT-NVR approach to address the challenge of real-time vehicle routing in dynamic environments. An encoder-decoder model with state broadcasting is trained via VCPO through view comparison; SCGS injects spatiotemporal compatibility into decoding to sample promising actions; and DCSM uses vehicle load peak and maximum slack time to mask infeasible decisions. Experiments show that VCPO improves training performance and reduces real-time operational cost, SCGS improves sampling efficiency, and DCSM assists the model in reducing cost and avoiding delay propagation. Further analyses reveal that future decision flexibility is a bottleneck and that the training trajectory evolves along a low-dimensional manifold. This work provides a reusable framework with offline training and online deployment for real-time vehicle routing. Future work will refine the model architecture and training method to enhance real-time cross-scale generalization, and investigate fine-tuning methods to improve future decision flexibility.

AI USE STATEMENT

In this work, we used Claude Code and DeepSeek-V4 for text polishing, format correction, reference proofreading, and code debugging. We have not used AI tools for method design, theoretical derivation, and experimental implementation. We have reviewed all AI-assisted works. LLM-suggested texts and edits were checked against the experimental results and the final manuscript was verified by the authors. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the source code, including dataset generation, offline training, online deployment, and evaluation scripts, in the supplementary material; it will be open-sourced on GitHub upon acceptance. Detailed descriptions of the model architecture, dataset creation, offline training and online deployment procedures are provided in the main paper and appendix.

REFERENCES

Aldair Alvarez, Jean-François Cordeau, and Raf Jans. The consistent vehicle routing problem with stochastic customers and demands. *Transportation Research Part B: Methodological*, 186:102968, 2024.

Wadi Khalid Anuar, Lai Soon Lee, Hsin-Vonn Seow, and Stefan Pickl. A multi-depot dynamic vehicle routing problem with stochastic road capacity: An MDP model and dynamic policy for post-decision state rollout algorithm in reinforcement learning. *Mathematics*, 10(15):2699, 2022.

Claudia Archetti, Leandro C Coelho, Maria Grazia Speranza, and Pieter Vansteenwegen. Beyond fifty years of vehicle routing: Insights into the history and the future. *European Journal of Operational Research*, 330(2):355–372, 2026.

Ahmad Bdeir, Jonas K Falkner, and Lars Schmidt-Thieme. Attention, filling in the gaps for generalization in routing problems. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2022.

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

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.

Nenad Bjelić, Milorad Vidović, Dražen Popović, and Branislava Ratković. Rolling-horizon approach in solving dynamic multisize multi-trailer container drayage problem. *Expert Systems with Applications*, 201:117170, 2022.

Yujie Chen, Biao Yuan, Yinzhi Zhou, Yuwei Chen, and Haoyuan Hu. Smart parcel consolidation at Cainiao. *INFORMS Journal on Applied Analytics*, 54(5):417–430, 2024.

K. T. Chung, C. K. M. Lee, and Y. P. Tsang. Neural combinatorial optimization with reinforcement learning in industrial engineering: a survey. *Artificial Intelligence Review*, 58(5):130, 2025.

Mustafa Demirbilek. A-static-periodic solution strategy for dynamic vehicle routing problem with simultaneous pickup and delivery. *Acta Infologica*, 5(1):1–12, 2021.

Han Fang, Zhihao Song, Paul Weng, and Yutong Ban. INViT: A generalizable routing problem solver with invariant nested view transformer. In *International Conference on Machine Learning (ICML)*, 2024.

Christian Friedrich and Ralf Elbert. Adaptive large neighborhood search for vehicle routing problems with transshipment facilities arising in city logistics. *Computers & Operations Research*, 137: 105491, 2022.

Shubhechyya Ghosal, Chin Pang Ho, and Wolfram Wiesemann. A unifying framework for the capacitated vehicle routing problem under risk and ambiguity. *Operations Research*, 72(2): 425–443, 2024.

Ahmed Heakl, Yahia Salaheldin Shaaban, Salem Lahlou, Martin Takáč, and Zangir Iklassov. SVRP-Bench: A realistic benchmark for stochastic vehicle routing problem. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Keld Helsgaun. An extension of the Lin-Kernighan-Helsgaun TSP solver for constrained traveling salesman and vehicle routing problems. Technical report, Roskilde University, 2017.

Waldy Joe and Hoong Chuin Lau. Deep reinforcement learning approach to solve dynamic vehicle routing problem with stochastic customers. In *International Conference on Automated Planning and Scheduling (ICAPS)*, pp. 394–402, 2020.

Minsu Kim, Junyoung Park, and Jinkyoo Park. Sym-NCO: Leveraging symmetricity for neural combinatorial optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Wouter Kool, Herke Van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations (ICLR)*, 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 (NeurIPS)*, 2020.

Sirui Li, Zhongxia Yan, and Cathy Wu. Learning to delegate for large-scale vehicle routing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Xiang Li, Xianzhe Wang, and Ziyan Feng. Dynamic repositioning in bike-sharing systems with uncertain demand: An improved rolling horizon framework. *Omega*, 126:103047, 2024.

Yile Liang, Haocheng Luo, Haining Duan, Donghui Li, Hongsen Liao, Jie Feng, Jiuxia Zhao, Hao Ren, Xuetao Ding, Ying Cha, Qingte Zhou, Chenqi Situ, Jinghua Hao, Ke Xing, Feifan Yin, Renqing He, Yang Sun, Yueqiang Zheng, Yipeng Feng, Zhizhao Sun, Jingfang Chen, Jie Zheng, and Ling Wang. Meituan's real-time intelligent dispatching algorithms build the world's largest minute-level delivery network. *INFORMS Journal on Applied Analytics*, 54(1):84–101, 2024.

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 (NeurIPS)*, 2023.

Fu Luo, Xi Lin, Yaoxin Wu, Zhenkun Wang, Xialiang Tong, Mingxuan Yuan, and Qingfu Zhang. Boosting neural combinatorial optimization for large-scale vehicle routing problems. In *International Conference on Learning Representations (ICLR)*, 2025.

Yi Ma, Xiaotian Hao, Jianye Hao, Jiawen Lu, Xing Liu, Xialiang Tong, Mingxuan Yuan, Zhigang Li, Jie Tang, and Zhaopeng Meng. A hierarchical reinforcement learning based optimization framework for large-scale dynamic pickup and delivery problems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Vinícius R Máximo and Mariá C V Nascimento. A hybrid adaptive iterated local search with diversification control to the capacitated vehicle routing problem. *European Journal of Operational Research*, 294(3):1108–1119, 2021.

Matheus J. Ota and Ricardo Fukasawa. Hardness of pricing routes for two-stage stochastic vehicle routing problems with scenarios. *Operations Research*, 73(4):2177–2187, 2025.

Victor Pillac, Michel Gendreau, Christelle Guéret, and Andrés L Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, 2013.

Rafael Praxedes, Teobaldo Bulhões, Anand Subramanian, and Eduardo Uchoa. A unified exact approach for a broad class of vehicle routing problems with simultaneous pickup and delivery. *Computers & Operations Research*, 162:106467, 2024.

Harilaos N Psaraftis, Min Wen, and Christos A Kontovas. Dynamic vehicle routing problems: Three decades and counting. *Networks*, 67(1):3–31, 2016.

Y D Shen, L W Peng, and J P Li. An improved estimation of distribution algorithm for multi-compartment electric vehicle routing problem. *Journal of Systems Engineering and Electronics*, 32(2):365–379, 2021.

Thibaut Vidal. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. *Computers & Operations Research*, 140:105643, 2022.

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.

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

W. Wang, Y. Shen, Y. Li, and Y. Fan. Multi consensus based estimation of distribution algorithm with domain knowledge for multi objective green vehicle routing. *IEEE Transactions on Evolutionary Computation*, 2026. doi: 10.1109/TEVC.2026.3704658. early access.

Xuan Wu, Di Wang, Lijie Wen, Yubin Xiao, Chunguo Wu, Yuesong Wu, Chaoyu Yu, Douglas L Maskell, and You Zhou. Neural combinatorial optimization algorithms for solving vehicle routing problems: A comprehensive survey with perspectives. *arXiv preprint arXiv:2406.00415*, 2024.

Zikang Yu, Jinbiao Chen, and Jiahai Wang. Combination-of-experts with knowledge sharing for cross-task vehicle routing problems. In *International Conference on Learning Representations (ICLR)*, 2026.

Chao Zhang, Jinbiao Xing, Hui Wang, Kuangzheng Li, Weihao Jiang, and Shiliang Pu. Adaptive distance-aware attention models for routing problems. In *International Joint Conference on Neural Networks (IJCNN)*, 2024.

Zhiqin Zhang, Yining Ma, Zhiguang Cao, and Hoong Chuin Lau. Probing neural combinatorial optimization models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.