Synergizing Imitation and Reinforcement Learning for the Periodic Capacitated Vehicle Routing Problem: A Comprehensive Research Report

1. Introduction: The Complexity of Temporal Logistics

The optimization of logistics and transportation networks serves as the backbone of the modern global economy. Within this domain, the Vehicle Routing Problem (VRP) has stood for over six decades as a central pillar of Operations Research (OR), challenging mathematicians and engineers to derive efficient routes for vehicle fleets serving geographically dispersed customers. However, as supply chains have evolved from static, single-day operations into dynamic, multi-echelon systems, the classical formulations of VRP have proven insufficient to capture the temporal complexities of real-world service level agreements. The Periodic Capacitated Vehicle Routing Problem (PCVRP) emerges from this necessity, extending the routing horizon from a single operational day to a tactical planning period—typically a week or a month—where customers require repeated visits according to specific frequency constraints and allowable service patterns.

The PCVRP represents a significant leap in combinatorial complexity compared to the standard Capacitated VRP (CVRP). It couples two distinct NP-hard problems: the tactical assignment problem, which involves selecting a valid visit schedule (pattern) for each customer to balance workload across the planning horizon, and the operational routing problem, which requires solving a CVRP for each day of that horizon. This hierarchical dependency creates a search space where local decisions—such as assigning a customer to a "Monday-Thursday" pattern versus a "Tuesday-Friday" pattern—can have non-linear, cascading effects on the feasibility and cost of the daily routes. A suboptimal pattern assignment can force vehicles to traverse the entire service territory on a specific day to serve a sparse set of widely scattered customers, neutralizing any efficiencies gained by advanced routing algorithms.[1]

Historically, the resolution of PCVRP has been the domain of exact methods, such as Branch-and-Cut or Branch-and-Price, and advanced metaheuristics like Hybrid Genetic Search (HGS) and Adaptive Large Neighborhood Search (ALNS). While exact methods offer guarantees of optimality, they are computationally intractable for instances exceeding a few dozen customers. Metaheuristics, conversely, scale to larger instances but often require substantial computational time to converge and rely on hand-crafted operators that do not adapt to the specific statistical distribution of the problem instances at hand.[3]

In response to these limitations, the field is witnessing a paradigm shift toward Neural Combinatorial Optimization (NCO). This approach leverages Deep Learning (DL) to approximate the solution of NP-hard problems. However, purely constructive Neural approaches (e.g., end-to-end Reinforcement Learning) often struggle with the hard constraints and long horizons of PCVRP. Consequently, a hybrid methodology is gaining prominence: the integration of Imitation Learning (IL) and Reinforcement Learning (RL). This report provides an exhaustive analysis of this convergence. It explores how Imitation Learning enables neural models to "clone" the structural wisdom of state-of-the-art heuristics like HGS, providing a stable initialization for Reinforcement Learning agents that subsequently refine these policies through trial-and-error interaction with the environment. By synthesizing insights from recent literature on Imitation Improvement Learning (IIL), Hierarchical RL, and Neural-Guided Metaheuristics, this document charts the trajectory of next-generation solvers that promise the speed of neural inference with the precision of classical optimization.

2. Theoretical Foundations of the Periodic Capacitated VRP

To understand the mechanisms by which machine learning can be applied to PCVRP, one must first establish a rigorous understanding of the problem's mathematical structure and the inherent constraints that differentiate it from standard routing tasks.

2.1 Mathematical Formulation and Decision Hierarchy

The PCVRP is defined on a complete undirected graph $G = (V, E)$, where $V = \{0, 1,..., n\}$ represents the set of vertices, with node 0 denoting the central depot and nodes $V_c = \{1,..., n\}$ representing the customers. The set $E = \{(i, j) : i, j \in V, i \neq j\}$ represents the edges connecting these nodes, each associated with a travel cost c_{ij}, typically corresponding to Euclidean distance or travel time.[5]

The temporal dimension is introduced through a planning horizon $\mathcal{T} = \{1,..., T\}$, representing the days over which services must be scheduled. Each customer i has a service frequency f_i, indicating the number of times they must be visited within the horizon $\mathcal{T}$. Crucially, these visits cannot occur arbitrarily; they must adhere to a set of allowable visit patterns (or schedules) R_i. A pattern $p \in R_i$ is a subset of $\mathcal{T}$ such that $|p| = f_i$. For instance, if $T=5$ (Monday to Friday) and a customer requires two visits ($f_i=2$), valid patterns might include $\{1, 4\}$ (Monday-Thursday) or $\{2, 5\}$ (Tuesday-Friday).

The optimization task involves three simultaneous decisions:

1. **Pattern Selection:** Choosing exactly one pattern $p \in R_i$ for each customer i.
2. **Fleet Assignment:** Determining which vehicle k from a homogeneous fleet K serves customer i on day $t \in p$.
3. **Route Sequencing:** Determining the order of visits for each vehicle on each day to

minimize cost.

This structure can be formalized using binary decision variables. Let y_{ip} be equal to 1 if pattern p is selected for customer i, and 0 otherwise. Let x_{ijk}^t be equal to 1 if vehicle k travels from node i to node j on day t. The objective is to minimize the total travel cost over the horizon:

$$\text{Minimize } Z = \sum_{t \in \mathcal{T}} \sum_{k \in K} \sum_{(i, j) \in E} c_{ij} x_{ijk}^t$$
Subject to constraints ensuring that:

- Each customer is assigned exactly one valid pattern: $\sum_{p \in R_i} y_{ip} = 1, \forall i \in V_c$.
- Customers are visited on every day belonging to their chosen pattern.
- Vehicle capacity Q is not exceeded on any route on any day: $\sum_{i \in V_c} q_i \sum_{j \in V} x_{ijk}^t \leq Q, \forall k \in K, \forall t \in \mathcal{T}$, where q_i is the demand of customer i.[6]

2.2 The Challenge of Interdependency

The core difficulty of PCVRP lies in the interdependence of the daily subproblems. In a standard VRP, the constraints are spatial. In PCVRP, they are spatio-temporal. A decision to visit Customer A on Monday (as part of a Monday-Thursday pattern) consumes vehicle capacity on Monday, potentially forcing Customer B (who is geographically close to A) to be serviced by a second vehicle if Monday's fleet capacity is saturated. This coupling means that the "goodness" of a pattern assignment cannot be evaluated in isolation; it depends on the assignments of all other customers.

This structure poses a severe challenge for pure Reinforcement Learning approaches. An RL agent that constructs a solution sequentially might assign patterns to the first 90% of customers that look spatially efficient, only to find that the remaining 10% cannot be inserted into any valid route on the remaining days due to capacity violations. This "sparse reward" problem—where the infeasibility signal is received only after thousands of decisions—is a primary motivator for integrating Imitation Learning, which can guide the agent toward feasible regions of the search space.[8]

2.3 Benchmark Instances and Data Characteristics

The evaluation of PCVRP algorithms relies on standardized benchmark sets that encode these complexities.

- **Christofides and Beasley (1984):** The seminal dataset comprising instances derived from classical VRP problems but augmented with period and frequency data. These instances define specific allowed pattern sets (e.g., Pattern A: Days 1, 3; Pattern B: Days 2, 4) and are relatively small by modern standards (50–100 customers).[1]

- **Cordeau et al. (1997):** A more extensive set of larger instances (up to several hundred customers) that introduced tighter capacity and duration constraints. These instances are critical for evaluating the scalability of modern neural solvers.[10]
- **Vidal et al. (2012):** Instances designed to test the limits of Hybrid Genetic Search, often featuring complex mixes of frequencies and highly constrained pattern sets.[4]

Understanding the file formats and geometric properties of these instances is crucial for training learning models. For example, Cordeau's instances specify the location coordinates (x, y), service duration d, demand q, frequency f, and the list of valid visit combinations for each customer. A neural network must learn to encode not just the spatial coordinates (as in TSP) but also the discrete combinatorial structure of the list of valid patterns.[10]

3. Classical Heuristics: The "Experts" for Imitation

To implement Imitation Learning, one requires an "expert" oracle capable of generating high-quality training data. In the context of PCVRP, the gold standard for decades has been the class of metaheuristics, specifically Evolutionary Algorithms and Local Search variants. Understanding how these experts function is essential for designing neural networks that can effectively clone their behavior.

3.1 Hybrid Genetic Search (HGS)

The Hybrid Genetic Search (HGS), extensively developed by Vidal et al., represents the state-of-the-art for PCVRP. It is a "memetic" algorithm, meaning it combines the global exploration of a genetic algorithm with the local exploitation of individual learning (local search).[4]

Mechanism of HGS:

1. **Giant Tour Representation:** HGS often avoids explicit day-assignment in its chromosome representation. Instead, it represents the solution as a "Giant Tour" (a permutation of all customer visits over the horizon) and uses a Split algorithm (dynamic programming) to optimally partition this giant tour into feasible daily routes. This reduces the problem search space significantly.
2. **Pattern Processing:** For PCVRP, HGS extends the chromosome to include pattern assignments. The algorithm evolves both the sequence of visits and the pattern choices simultaneously.
3. **Local Search Operators:** The critical component for Imitation Learning is the local search phase. HGS employs operators like **2-opt** (reversing a subsequence), **Or-opt** (moving a subsequence), and **SWAP*** (an inter-route exchange operator). SWAP* is particularly powerful as it can move a customer from a route on Day 1 to a route on Day 2, implicitly changing the customer's pattern assignment if necessary.[13]
4. **Diversity Management:** HGS actively manages population diversity to prevent

premature convergence, a feature that RL agents effectively try to replicate via entropy regularization in their policy updates.

The significance of HGS for machine learning is twofold:

- **Data Generation:** HGS can generate millions of state-action pairs (e.g., "In state S, the operator SWAP* was applied to edges e_1 and e_2"). This data forms the training set for Behavior Cloning.
- **Baseline Performance:** HGS provides the benchmark "Best Known Solutions" (BKS) that any hybrid IL/RL model must aim to match or approximate with lower latency.[14]

3.2 LKH-3: The Lin-Kernighan-Helsgaun Heuristic

While HGS dominates in solution quality for many variants, the LKH-3 solver extends the legendary Lin-Kernighan TSP heuristic to constrained vehicle routing, including PCVRP. LKH-3 operates by transforming the VRP into a Traveling Salesman Problem (TSP) with specific constraints and solving it using sophisticated k-opt moves (where k can be dynamic). LKH-3 is notable for its speed and robustness on specific instance types. In hybrid learning frameworks like **NeuroLKH**, the neural network is trained to predict which edges LKH-3 is likely to select, allowing the heuristic to ignore unpromising edges and run significantly faster.[9]

3.3 The Computational Bottleneck

Despite their effectiveness, classical experts like HGS and LKH-3 suffer from a fundamental limitation: **computational latency**. Solving a 1,000-customer PCVRP instance to near-optimality with HGS can take minutes or even hours of CPU time. Furthermore, these algorithms are "blind" to the distribution of the problem instances; they start their search from scratch for every new instance, failing to leverage the fact that real-world logistics networks often look similar day-to-day (e.g., the depot is always in the center, and customers are clustered in suburbs). This inability to "learn" is precisely the gap that combined Imitation and Reinforcement Learning seeks to fill.[16]

4. Neural Combinatorial Optimization: Architectures for Routing

Before delving into the hybrid training strategies, it is necessary to detail the neural architectures that serve as the "brains" of these agents. Unlike image processing (CNNs) or natural language (RNNs/Transformers), routing problems reside on graph structures, requiring specialized architectures.

4.1 Graph Neural Networks (GNNs)

Graph Neural Networks are the dominant architecture for NCO. They operate by maintaining a feature vector (embedding) for each node (customer) and updating these vectors by

aggregating information from neighboring nodes.

Application to PCVRP:

- **Input Features:** Each node i is initialized with a feature vector $h_i^{(0)} = [x_i, y_i, q_i, f_i, \text{PatternEncoding}_i]$.
- **Message Passing:** In each layer l, node i gathers messages from its neighbors. For PCVRP, "neighbors" might be defined spatially (k-nearest neighbors) or implicitly via an attention mechanism.
- **Pattern Encoding:** A critical innovation for PCVRP is how to encode the list of allowable patterns. Recent approaches use **multi-hot encoding** or learned embeddings to represent the set of feasible days. For example, if the horizon is 5 days, a pattern $\{1,3\}$ might be encoded as $$. The GNN must learn that customers with compatible patterns (e.g., overlapping days) compete for capacity.[10]

4.2 Attention Models and Transformers

The **Attention Model (AM)**, popularized by Kool et al. (2019), treats the VRP as a sequence-to-sequence translation problem.

- **Encoder:** A Transformer encoder processes the set of customers to produce contextualized embeddings.
- **Decoder:** An autoregressive decoder generates the solution one step at a time. At each step, it outputs a probability distribution over the available nodes, deciding which customer to visit next.

The PCVRP Adaptation:
For PCVRP, the decoder's complexity increases. It must not only decide who to visit next but also which day to assign. Hierarchical architectures often use one Attention model to assign patterns (creating sub-problems) and another to route the vehicles for each day.

- **Anisometric Encodings:** Recent research into **HADES** and other advanced NCO models highlights the need for "anisometric" positional encodings. Unlike text where position is sequential (1, 2, 3...), VRP nodes exist in 2D space. For PCVRP, the encoding must be **Spatio-Temporal**, capturing that two customers are "close" if they are geographically nearby *and* require visits on the same days.[19]

4.3 Policy Networks for Improvement

In Improvement Learning (as opposed to Constructive Learning), the neural network inputs a *complete solution* and outputs a *modification*.

- **Input:** A graph representing the current routes and pattern assignments.
- Output: A probability map over edges or nodes, suggesting a local search move (e.g., "Swap node i and node j").
  This architecture is often lighter than constructive models because it only needs to recognize local inefficiencies (like crossing edges) rather than plan a global tour from

scratch.[20]

5. Imitation Learning: Cloning the Expert

Imitation Learning (IL) acts as the bridge between classical OR and modern AI. In the context of PCVRP, the search space is too vast for an RL agent to explore efficiently from a random initialization. The probability of randomly selecting a valid combination of patterns and routes that satisfies all capacity and frequency constraints is infinitesimally small. IL provides the necessary "warm start."

5.1 Behavior Cloning (BC) in Routing

Behavior Cloning is the simplest form of IL, treating the problem as Supervised Learning.

1. **Data Collection:** Run an expert solver (e.g., HGS or LKH-3) on a large set of random PCVRP instances. Record the solutions. For constructive approaches, record the sequence of node selections. For improvement approaches, record the sequence of local search moves.
2. Training: Train the neural network π_θ to minimize the cross-entropy loss between its predicted action probabilities and the expert's actions.

   $$\mathcal{L}_{BC}(\theta) = - \sum_{(s, a^*) \in \mathcal{D}} \log \pi_\theta(a^* | s)$$

   Where a^* is the action taken by the expert in state s.

Limitations in PCVRP:
While BC can teach the network the basic "rules" (e.g., don't exceed capacity, respect time windows), it suffers from covariate shift. If the network makes a small error during inference, it enters a state distribution it never saw during expert training (since the expert never makes mistakes). In PCVRP, a single bad pattern assignment can lead to a state where no feasible routing is possible—a situation the expert HGS would never encounter.

5.2 Imitation Improvement Learning (IIL)

To address the limitations of pure BC, recent research has coalesced around **Imitation Improvement Learning (IIL)**. This framework, highlighted in studies on large-scale routing [8], is specifically designed to combine the robustness of heuristics with the learnability of neural networks.

The IIL Mechanism:

1. **Expert-Guided Perturbation:** Instead of just recording the expert's optimal path, the training process involves *perturbing* the expert's solution (e.g., randomly swapping patterns) to create suboptimal states.
2. **Repair Learning:** The expert solver is then asked to "repair" this suboptimal state. The neural network learns this repair operation. This teaches the network how to recover

from mistakes, directly addressing covariate shift.

3. **Dual-Phase Training:**
    - *Phase 1 (Imitation):* The network minimizes the Kullback-Leibler (KL) divergence between its policy and the expert's repair moves.
    - *Phase 2 (Self-Improvement):* The network switches to an RL objective (see Section 6) to refine the policy further.

Relevance to PCVRP:
In PCVRP, IIL is particularly potent for the Pattern Assignment subproblem. An expert like HGS might use a complex evaluation function to decide that moving Customer X from "Mon-Wed" to "Tue-Thu" is beneficial. An IIL agent learns to predict this swap based on features like "Monday Capacity Utilization" and "Tuesday Cluster Centroid," effectively distilling the complex logic of HGS into a fast forward-pass of a GNN.8

6. Reinforcement Learning: Optimizing Beyond the Expert

While Imitation Learning provides a solid foundation, it theoretically caps the agent's performance at the level of the expert. Reinforcement Learning (RL) allows the agent to transcend the expert by optimizing the objective function directly.

6.1 The Markov Decision Process (MDP) for PCVRP

Formalizing PCVRP as an RL problem requires a careful definition of the MDP tuple (S, A, P, R).

- **State (S_t):** The state must capture both the static graph features and the dynamic solution status.
    - *Static:* Customer locations, demands q_i, frequencies f_i, allowed patterns R_i.
    - *Dynamic:* Current pattern assignments for all customers, current routes for each day, remaining capacity of vehicles on each day.
- **Action (A_t):**
    - *Constructive RL:* Select the next customer to add to the current route on the current day.
    - *Improvement RL:* Select a "Destroy" operator (e.g., remove 10 customers) and a "Repair" operator (e.g., re-insert them using a greedy heuristic), or select a specific node swap.
- Reward (R_t): The reward is typically the negative change in total cost.

  $$R_t = - (Cost(S_{t+1}) - Cost(S_t))$$

  To handle constraints, penalty methods are often used. If an action leads to a capacity violation on Day 3, a large negative reward is administered. Alternatively, masking is used

to prevent the agent from selecting invalid actions (e.g., the output probability for an invalid pattern is set to zero).17

6.2 Policy Gradient Methods

The most common RL algorithms for routing are **Policy Gradient** methods, such as REINFORCE with baseline or Proximal Policy Optimization (PPO).

The Objective:
Maximize the expected reward: $J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}$.
The gradient is estimated as:

$$\nabla J(\theta) \approx \frac{1}{N} \sum_{i=1}^N (R(\tau_i) - b(s_i)) \nabla \log \pi_\theta(\tau_i)$$

where $b(s_i)$ is a baseline (often the value of a greedy rollout or a Critic network) used to reduce variance.
Combined IL/RL Training:
In the combined framework, the loss function is a weighted sum:

$$\mathcal{L}_{total} = \lambda_{IL} \mathcal{L}_{BC} + \lambda_{RL} \mathcal{L}_{PG}$$

Initially, λ_{IL} is high, forcing the agent to mimic the expert. As training progresses, λ_{IL} decays and λ_{RL} increases, allowing the agent to explore and optimize the reward directly. This curriculum learning strategy is essential for PCVRP, as it prevents the agent from getting stuck in local optima during the early phases when it hasn't yet learned valid pattern structures.22

6.3 Handling Sparse Rewards in Periodic Settings

In PCVRP, the quality of a pattern assignment is only revealed after the routing phase is complete. This delayed reward makes credit assignment difficult. To mitigate this, **Hierarchical RL (HRL)** is employed.

- **Manager Agent:** Assigns patterns. It receives a reward based on the *estimated* cost of the resulting daily VRPs (often approximated by a fast heuristic or a learned value function).
- Worker Agent: Solves the daily VRPs. It receives immediate rewards for minimizing route distance.
  This decomposition decouples the temporal assignment from the spatial routing, allowing both agents to learn more efficiently.9

7. Integrated Frameworks: The State of the Art

The synthesis of IL and RL has given rise to several distinct frameworks that represent the cutting edge of PCVRP research.

7.1 Hierarchical Learning-based Graph Partition (HLGP)

The HLGP framework [9] explicitly addresses the scalability of VRPs through a divide-and-conquer approach.

- **Decomposition:** A high-level policy (trained via RL) partitions the large graph into smaller subproblems. For PCVRP, this partition is both spatial (clustering customers) and temporal (assigning days).
- **Expert Guidance:** The decomposition policy is initially trained via IL to mimic a high-quality partitioner (like the sweeping algorithm or METIS), ensuring the subproblems are balanced and solvable.
- **Subproblem Solving:** The resulting smaller VRPs are solved by a low-level neural solver or a classical heuristic.
- **Result:** This approach allows neural methods to tackle PCVRP instances with thousands of nodes, a scale where flat RL models fail completely.

7.2 NeuroLKH: Neural-Guided Metaheuristics

NeuroLKH [9] represents a hybrid approach where the neural network does not replace the solver but *augments* it.

- **Concept:** The LKH-3 heuristic is powerful but slow because it searches a vast neighborhood of possible edge swaps. NeuroLKH uses a GNN to predict the probability that any given edge (i, j) exists in the optimal solution.
- **IL Component:** The GNN is trained via Supervised Learning on a dataset of optimal solutions (or best-known solutions from HGS).
- **Integration:** During inference, LKH-3 is restricted to search only the "promising" edges identified by the GNN (sparsification).
- **Impact on PCVRP:** This is particularly effective for periodic problems. The GNN can learn that an edge between Customer A and Customer B is only promising if they are assigned to the same day. By filtering out incompatible edges, NeuroLKH dramatically accelerates the operational routing phase of PCVRP.[9]

7.3 Imitation-Improvement for Large Scale (IIL)

As detailed in Section 5.2, **IIL** [8] focuses on iteratively improving a solution.

- **Clockwise Clustering:** To handle large-scale PCVRP, Bui and Mai propose a "Clockwise Decomposition" where nodes are clustered angularly.
- **Learned Operators:** The agent learns to apply operators like 2-opt or node-relocation

within these clusters.
- Expert Teacher: The HGS algorithm serves as the teacher. The agent learns to mimic HGS's decisions on which cluster to optimize and which operator to apply.
  This method has achieved new state-of-the-art results on large-scale benchmarks, proving that learning to improve is more scalable than learning to construct.26

8. Software Implementation and Benchmarking

The transition from theoretical frameworks to working solvers relies on robust software ecosystems and standardized benchmarks.

8.1 PyVRP: The Engine for Hybrid Solvers

PyVRP [14] has emerged as a critical tool for this research. It is a high-performance Python wrapper around a C++ implementation of the Hybrid Genetic Search (HGS).

- **Architecture:** PyVRP exposes the internal components of HGS (Population, Crossover, Local Search) to Python. This allows researchers to inject Neural Network models directly into the search loop. For example, one could replace the standard "Survivor Selection" logic of the Genetic Algorithm with a learned ranking model trained via RL.
- **Periodic Support:** While PyVRP natively focuses on CVRP and VRPTW, its modular design allows for the implementation of PCVRP. Researchers can model PCVRP by extending the ProblemData class to include pattern constraints and modifying the CostEvaluator to penalize pattern violations. The active GitHub discussions [29] indicate that native PCVRP support is a highly requested feature, often implemented by users via custom extensions.
- **Role in Hybrid Learning:** PyVRP acts as the *environment* for the RL agent. The agent (Python/PyTorch) manipulates the solution, and PyVRP (C++) rapidly evaluates the cost and feasibility, providing the reward signal.

8.2 Benchmarking and Performance Analysis

Validating hybrid solvers requires rigorous testing against established datasets.

- **Christofides and Beasley (1984):** These 22 instances are the standard for PCVRP. They range from 50 to 100 customers with varying tightness of capacity and frequency constraints. Hybrid methods like IIL have shown the ability to match the BKS (Best Known Solutions) on these small instances with inference times under 1 second.[31]
- **Cordeau et al. (1997):** A set of larger, more difficult instances. Pure RL approaches often fail here due to the tight coupling of constraints. Hybrid approaches (NeuroLKH) demonstrate their value here, achieving gaps of <1% to BKS while running 10-100x faster than pure HGS.[10]
- **VRP-REP:** The central repository for VRP solution data.[33] It tracks the BKS for all variants. For PCVRP, the BKS are almost exclusively held by HGS-variants (Vidal) or LKH-3. The goal of current NCO research is not necessarily to beat these BKS in quality (which is

near-optimal) but to achieve comparable quality in a fraction of the time (real-time solving).

Table 1: Comparative Analysis of Solution Methodologies for PCVRP

Feature	Pure Metaheuristic (HGS)	Pure DRL (AM/POMO)	Combined IL + RL (IIL/NeuroLKH)
Training Time	None	High (Hours/Days)	Very High (Days)
Inference Time	High (Minutes to Hours)	Low (Milliseconds)	Medium (Seconds)
Solution Quality	State-of-the-Art (Optimal/Near-Optimal)	Good on small, poor on large	Competitive with SOTA
Scalability	Good (with time)	Poor (>100 nodes)	Excellent (via Decomposition)
Periodic Handling	Native (Genetic Operators)	Difficult (State complexity)	Learned (Hierarchical)
Generalization	Perfect (Instance-agnostic)	Poor (Distribution-specific)	Improved (via Expert Anchoring)

9. Insights and Future Directions

The integration of Imitation and Reinforcement Learning for PCVRP is not merely a technical exercise; it represents a fundamental shift in how we approach NP-hard problems in logistics.

Second-Order Insight: The "Black Box" Paradox
While combined approaches like NeuroLKH show immense promise, they introduce opacity. Classical heuristics are interpretable (e.g., "swap these two nodes to untangle the route"). Neural operators act as "black boxes" that suggest moves based on high-dimensional feature correlations. The success of these methods implies that PCVRP solutions contain latent structural patterns—complex geometric and temporal correlations—that are too subtle for human algorithm designers to codify but are accessible to GNNs. Future research will likely

focus on "Explainable NCO," reverse-engineering the moves learned by the RL agent to discover new classical heuristics.34

Third-Order Insight: From Solver Design to Data Generation

The heavy reliance on Imitation Learning shifts the research burden from algorithm design to data generation. The performance of an IIL agent is strictly capped by the quality and diversity of the expert data it consumes. This is driving a trend toward "Data-Centric AI" in Operations Research. Researchers are now focusing on generating synthetic PCVRP instances that cover "corner cases" (e.g., extremely clustered customers, highly restrictive patterns) to train more robust agents. If the training data only comes from one type of solver (e.g., HGS), the neural agent risks overfitting to the biases of that solver, potentially blinding it to alternative solution strategies.

Future Trajectories:

1. **LLM-Driven Heuristics:** Emerging work suggests using Large Language Models (LLMs) not just to solve the problem, but to *write the code* for the heuristics used by the RL agent. An LLM could analyze a specific PCVRP instance and generate a custom Python operator for PyVRP to use in the local search phase.[15]
2. **Hardware-Accelerated Solvers:** With frameworks like PyVRP optimizing the C++ backend and NCO models leveraging GPUs, we are approaching a "Neural-Solver Stack" where the entire optimization pipeline runs on the GPU. This could enable "Instant Logistics," where massive multi-period schedules are re-optimized in real-time as new orders arrive.

10. Conclusion

The Periodic Capacitated Vehicle Routing Problem stands as a formidable challenge that resists simple solutions due to its entangled hierarchy of assignment and routing. The convergence of Imitation Learning and Reinforcement Learning offers a transformative path forward. By using Imitation Learning to distill the accumulated wisdom of decades of metaheuristic research (embodied in experts like HGS) and Reinforcement Learning to provide adaptive, instance-specific refinement, hybrid solvers are breaking the traditional trade-off between speed and quality.

Current evidence suggests that **Hierarchical Imitation Improvement Learning**—where RL manages high-level pattern assignment and IL-guided operators refine low-level routing—is the most promising architecture for scaling to real-world dimensions. Supported by robust software ecosystems like PyVRP and rigorous benchmarks, these "neural-symbolic" systems are poised to redefine the capabilities of computational logistics, offering solvers that are not only faster but fundamentally more adaptive to the dynamic nature of global supply chains. The future of PCVRP solution lies not in choosing between learning and search, but in their seamless, symbiotic integration.

Works cited

1. The Period Routing Problem | Request PDF - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/229744511_The_Period_Routing_Problem
2. A unified model framework for the multi-attribute consistent periodic vehicle routing problem, accessed December 29, 2025, https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0237014
3. Integrating Machine Learning Into Vehicle Routing Problem: Methods and Applications - IEEE Xplore, accessed December 29, 2025, https://ieeexplore.ieee.org/iel8/6287639/10380310/10583875.pdf
4. A Hybrid Genetic Algorithm for Multidepot and Periodic Vehicle Routing Problems, accessed December 29, 2025, https://www.researchgate.net/publication/230846309_A_Hybrid_Genetic_Algorithm_for_Multidepot_and_Periodic_Vehicle_Routing_Problems
5. New formulation and valid inequalities for a periodic capacitated vehicle routing problem with multiple depots, heterogeneous fleet, and hard time-windows - NIH, accessed December 29, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC12578217/
6. Branch-and-Price-and-Cut for the Periodic Vehicle Routing Problem with Flexible Schedule Structures, accessed December 29, 2025, https://logistik.bwl.uni-mainz.de/files/2018/12/PVRP_Paper_TR.pdf
7. solving a periodic capacitated vehicle routing problem using simulated annealing algorithm for, accessed December 29, 2025, https://bjopm.org.br/bjopm/article/download/866/920/7658
8. Imitation Improvement Learning for Large-scale Capacitated Vehicle Routing Problems | OpenReview, accessed December 29, 2025, https://openreview.net/forum?id=K5UfKyHIBS
9. Hierarchical Learning-based Graph Partition for Large-scale Vehicle Routing Problems | Request PDF - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/388954586_Hierarchical_Learning-based_Graph_Partition_for_Large-scale_Vehicle_Routing_Problems
10. Description for files of Cordeau's Instances | Vehicle Routing Problem - NEO, accessed December 29, 2025, https://neo.lcc.uma.es/vrp/vrp-instances/description-for-files-of-cordeaus-instances/
11. MDVRP-Instances/DESCRIPTION.md at master · fboliveira/MDVRP-Instances - GitHub, accessed December 29, 2025, https://github.com/fboliveira/MDVRP-Instances/blob/master/DESCRIPTION.md
12. A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems, accessed December 29, 2025, https://www.uv.es/route2011/Gendreau.pdf
13. (PDF) Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/356580708_Hybrid_genetic_search_for_the_CVRP_Open-source_implementation_and_SWAP_neighborhood
14. PyVRP: A High-Performance VRP Solver Package - ResearchGate, accessed

December 29, 2025, https://www.researchgate.net/publication/377789326_PyVRP_A_High-Performance_VRP_Solver_Package
15. VRPAgent: LLM-Driven Discovery of Heuristic Operators for Vehicle Routing Problems, accessed December 29, 2025, https://arxiv.org/html/2510.07073v1
16. Learning heuristic selection using a Time Delay Neural Network for Open Vehicle Routing, accessed December 29, 2025, https://www.researchgate.net/publication/316095940_Learning_heuristic_selection_using_a_Time_Delay_Neural_Network_for_Open_Vehicle_Routing
17. Leveraging Transfer Learning in Deep Reinforcement Learning for Solving Combinatorial Optimization Problems Under Uncertainty - IEEE Xplore, accessed December 29, 2025, https://ieeexplore.ieee.org/iel8/6287639/10380310/10766597.pdf
18. 1 Introduction - arXiv, accessed December 29, 2025, https://arxiv.org/html/2403.13795v2
19. Hierarchical Aggregation Deconstruction Search for Vehicle Routing Problems | OpenReview, accessed December 29, 2025, https://openreview.net/forum?id=NLgJcADMtr
20. "Imitation improvement learning for large-scale capacitated vehicle ..., accessed December 29, 2025, https://ink.library.smu.edu.sg/sis_research/8025/
21. (PDF) Leveraging Transfer Learning in Deep Reinforcement Learning for Solving Combinatorial Optimization Problems Under Uncertainty - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/386124480_Leveraging_Transfer_Learning_in_Deep_Reinforcement_Learning_for_Solving_Combinatorial_Optimization_Problems_Under_Uncertainty
22. Learning to Optimize Vehicle Routes Problem: A Two-Stage Hybrid Reinforcement Learning, accessed December 29, 2025, https://ieeexplore.ieee.org/document/10490595/
23. Hierarchical Learning-based Graph Partition for Large-scale Vehicle Routing Problems - arXiv, accessed December 29, 2025, https://arxiv.org/pdf/2502.08340?
24. Hierarchical Reinforcement Learning for Vehicle Routing Problems with Time Windows, accessed December 29, 2025, https://www.researchgate.net/publication/352724597_Hierarchical_Reinforcement_Learning_for_Vehicle_Routing_Problems_with_Time_Windows
25. Scalable Learning for Multi-Agent Route Planning: Adapting to Diverse Task Scales, accessed December 29, 2025, https://www.researchgate.net/publication/380659247_Scalable_Learning_for_Multi-Agent_Route_Planning_Adapting_to_Diverse_Task_Scales
26. Smart Maze Solver Using Reinforcement Learning | Request PDF - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/publication/396432158_Smart_Maze_Solver_Using_Reinforcement_Learning
27. Neural Genetic Search in Discrete Spaces | Request PDF - ResearchGate, accessed December 29, 2025,

https://www.researchgate.net/publication/389090136_Neural_Genetic_Search_in_Discrete_Spaces
28. PyVRP: a high-performance VRP solver package - arXiv, accessed December 29, 2025, https://arxiv.org/pdf/2403.13795
29. Common VRP variants · Issue #441 · PyVRP/PyVRP - GitHub, accessed December 29, 2025, https://github.com/PyVRP/PyVRP/issues/441
30. Issues · PyVRP/PyVRP - GitHub, accessed December 29, 2025, https://github.com/PyVRP/PyVRP/issues
31. Geographic Distribution of Libraries in Christofides and Beasley (1984) 50b - ResearchGate, accessed December 29, 2025, https://www.researchgate.net/figure/Geographic-Distribution-of-Libraries-in-Christofides-and-Beasley-1984-50b_fig1_220413114
32. A set-covering based heuristic algorithm for the periodic vehicle routing problem - NIH, accessed December 29, 2025, https://pmc.ncbi.nlm.nih.gov/articles/3990422/
33. Best Known Solutions - VRP-REP: the vehicle routing problem repository, accessed December 29, 2025, http://www.vrp-rep.org/solutions.html
34. 129 Machine Learning into Metaheuristics: A Survey and Taxonomy, accessed December 29, 2025, https://zeus.inf.ucv.cl/~bcrawford/MII918-ALGORITMOS-BIOINSPIRADOS/PAPERS/Machine%20Learning%20into%20Metaheuristics-Talbi-2021.pdf
35. VRPAgent: LLM-Driven Discovery of Heuristic Operators for Vehicle Routing Problems, accessed December 29, 2025, https://openreview.net/forum?id=02mBAZjFzp