logistics

MDPI

Article

AI-Driven Orchestration of AutoML-Based Demand Forecasting, Fleet Allocation, and Route Optimization in Last-Mile Delivery: A Single-Instance Case Comparison of Metaheuristic and Agentic AI Routing

Aymane Labtiti [1] and Jamal Benhra [2,*]

[1] ECC–UM6P MILEX Project (Ecole Centrale Casablanca, Mohammed VI Polytechnic University), Casablanca 20000, Morocco; aymane.labtiti@gmail.com
[2] Laboratory of Advanced Research in Industrial and Logistic Engineering (LARILE), National High School for Electricity and Mechanics (ENSEM), Hassan II University of Casablanca, Casablanca 20000, Morocco
* Correspondence: jamal.benhra@ensem.ac.ma

Abstract

Background: Last-mile delivery couples demand forecasting, heterogeneous fleet allocation, and routing under time windows, yet these decisions are usually studied in isolation. The object of this study is the forecast-to-route decision pipeline of an urban parcel operator; we examine the hypothesis that no single routing paradigm dominates once solvers are evaluated under one shared feasibility-aware objective. *Methods*: We implement an orchestration architecture in which an AutoML supervisor selects transfer-learned recurrent backbones per zone, forecasts become Vehicle Routing Problem with Time Windows instances over the Casablanca road network, and five solvers—three metaheuristics (GA, PSO, ACO) and two multi-agent reinforcement learning methods (MAPPO, MADQN)—minimize one common fitness function. The demand corpus is synthetic, generated from the LaDe Hangzhou partition to preserve its spatial and temporal structure within the Casablanca boundary. *Results*: Across five backbones, accuracy differences were small (all $R^2 \geq 0.845$; best bidirectional LSTM, $R^2 = 0.865$, wMAPE = 29.4%). On one 50-customer instance solved once per solver, GA returned the best feasible solution, whereas MAPPO reached the shortest distance only by violating time windows. *Conclusions*: Backbone choice is not the decisive forecasting decision, and constraint handling rather than paradigm separates the routing solvers. Multi-instance, budget-matched evaluation remains necessary and is specified.

Keywords: last-mile delivery; AutoML; agentic AI; multi-agent reinforcement learning; demand forecasting; vehicle routing problem with time windows

Check for updates

Academic Editors: Vasilis Zeimpekis and Michael Madas

Received: 2 July 2026
Revised: 22 August 2026
Accepted: 3 September 2026
Published: 9 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Last-mile delivery is widely recognized as one of the most costly, energy-intensive, and operationally complex stages of modern supply chains [1,2]. It frequently accounts for a disproportionate share of total distribution costs, driven by low drop densities, fragmented delivery windows, high vehicle idle times, and the growing volume of business-to-consumer parcels generated by e-commerce. These pressures are amplified in dense urban environments, where congestion, restricted access zones, heterogeneous vehicle fleets, and increasingly strict service-level and sustainability requirements interact in ways that are difficult to plan for using static or manually tuned strategies. Effective last-mile planning

therefore requires the simultaneous coordination of three tightly coupled decisions: accurate short-term demand forecasting, efficient fleet allocation across heterogeneous vehicle types, and adaptive vehicle routing under dynamic and uncertain operating conditions. Recent advances in artificial intelligence and intelligent transportation systems have accelerated the development of data-driven approaches supporting each of these decisions [3].

On the forecasting side, recurrent neural networks, attention mechanisms, and transfer learning have demonstrated strong performance for short-term demand prediction in transportation and logistics systems [4,5]. Automated Machine Learning (AutoML) frameworks have further improved model selection and hyperparameter optimization under heterogeneous operational conditions [6], and machine learning is now routinely embedded in supply-chain decision processes more broadly [7]. On the routing side, the Vehicle Routing Problem with Time Windows (VRPTW) remains the canonical formulation for last-mile route planning [8] and is most commonly addressed using metaheuristics such as Genetic Algorithms (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO), which offer strong solution quality for medium-scale instances at moderate computational cost [9–12]. More recently, reinforcement learning (RL) and multi-agent systems have introduced adaptive routing strategies capable of responding to dynamic demand and decentralized coordination requirements [13–17].

1.1. Gap Between Reported Findings and Operational Practice

The literature reports strong results for each stage separately, but three discrepancies persist between what is reported and what an operator can deploy. First, forecasting studies optimize statistical error metrics on held-out series, whereas the downstream consumer of a forecast is a fleet-sizing rule that needs a conservative bound rather than a conditional mean; the error metric that is reported is therefore not the quantity that governs operational cost. Second, routing studies report solution quality per instance family, but the solvers being compared are frequently allotted different search budgets and are rarely evaluated on the forecast-driven instances they would face in deployment. Third, and most consequentially, learning-based routing results are often reported in terms of distance or cumulative reward, whereas an operator is bound by service-level agreements; a policy that shortens routes by deferring time-constrained deliveries improves the reported metric while degrading the contracted service. Consequently, although each component is well studied, the composed pipeline—forecast, fleet, route—has rarely been implemented end-to-end and inspected for exactly these interface effects. Recent last-mile studies have begun to model the delivery system as a whole, through optimization of alternative delivery locations and heterogeneous customer preferences [18], cost-optimal coordination of automated delivery resources [19], and crowd-based fulfilment designs that reallocate capacity across the network [20], but these remain largely orthogonal to the question of which optimizer should run inside such a system.

1.2. Research Questions and Contributions

This study is positioned as an integration-and-diagnosis study rather than a solver-superiority study. Its novelty lies in placing forecasting, fleet allocation, and five structurally different routing strategies inside one executable architecture with a single shared objective function, and in reporting what that composition exposes—including effects that are invisible when the stages are evaluated separately. We address three research questions, each scoped to the evidence this study can supply:

- (RQ1) On hourly zone-level delivery demand, what accuracy does a diagnostic-driven AutoML forecasting layer with transfer-learned recurrent backbones attain, and how

much does the choice of recurrent backbone matter relative to the other design decisions in that layer?
- (RQ2) When classical metaheuristics (GA, PSO, ACO) and agentic MARL solvers (MAPPO, MADQN) are applied to the same forecast-driven VRPTW instance under one shared fitness function, what differences appear, and what do those differences reveal about how each paradigm handles feasibility?
- (RQ3) How can a solver-selection layer be specified so that heterogeneous solvers are interchangeable under a common instance descriptor, objective, and fallback contract—and what evidence would be required to establish that such a layer improves end-to-end performance?

RQ3 is deliberately posed as a design question. As detailed in Section 5.5, this study specifies and implements the selection layer but does not measure its benefit against an oracle or a best-fixed-solver baseline; that measurement requires a multi-instance corpus that this study does not have, and we treat it as the principal open item rather than as a demonstrated result.

The main contributions are summarized as follows:

- An executable orchestration architecture integrating demand forecasting, heterogeneous fleet allocation, and VRPTW routing, in which five structurally different solvers are interchangeable behind one instance descriptor and one shared fitness function, together with a fully documented data pipeline distinguishing what is real, transferred, and synthetic;
- A mechanistic diagnosis of the efficiency–feasibility contrast observed between metaheuristic and agentic solvers, tracing it to the interaction between step-wise action masking and post-hoc coverage repair rather than to the learning paradigm itself;
- A precise specification of the evaluations required to convert the architectural claims of this study into measured claims, including budget-matched comparison, multi-instance sampling, and orchestration regret against an oracle.

2. Literature Review

Recent advances in intelligent logistics systems have accelerated the integration of artificial intelligence techniques into last-mile delivery optimization. Existing studies mainly focus on demand forecasting, transfer learning, routing optimization, and intelligent orchestration mechanisms. This section reviews each strand and then isolates, in Section 2.4, the specific gap this study addresses.

2.1. Demand Forecasting and Transfer Learning

Accurate short-term demand forecasting is a prerequisite for effective fleet sizing and routing, since routing decisions can only be as good as the demand estimates that drive them. Deep learning architectures, particularly recurrent neural networks (RNNs) and attention-based models, have demonstrated strong performance for short-term demand forecasting in transportation systems, outperforming classical statistical baselines such as ARIMA and exponential smoothing when demand exhibits nonlinear temporal structure [4]. Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks address the vanishing-gradient limitations of vanilla RNNs and model long-range dependencies, while their bidirectional variants (BiLSTM, BiGRU) exploit both past and future context within the training window. A recurring difficulty in operational forecasting is data sparsity: newly opened zones, promotional periods, and low-volume regions provide too little history to train a reliable model from scratch. Transfer learning mitigates this by leveraging pretrained temporal representations learned on data-rich environments and fine-tuning them on target zones [5]. In parallel, AutoML frameworks automate model selection, feature construction,

and hyperparameter optimization, improving generalization while reducing the manual intervention required to deploy models across many zones [6]. Recurrent architectures have likewise proven effective for material and demand forecasting under limited historical data in manufacturing and logistics settings [21], and broader reviews document the steady integration of machine learning into supply-chain decision processes [7]. Taken together, these findings motivate a forecasting layer that combines recurrent backbones, transfer learning, and AutoML-based orchestration, which is the design adopted in this work.

2.2. Metaheuristic Routing Optimization

The VRPTW remains one of the most studied combinatorial optimization problems in logistics research. As an NP-hard extension of the classical vehicle routing problem, it cannot be solved to optimality for realistic instance sizes within practical time budgets, which explains the enduring reliance on heuristic and metaheuristic methods. Standardized instance families have long structured this literature, from the original Solomon benchmarks for time-windowed routing [8] to more recent large-scale capacitated instance sets designed to expose scaling behavior [22]. Classical metaheuristics such as GA, PSO, and ACO continue to demonstrate strong optimization capabilities for urban routing problems, each exploiting a different search paradigm: GA relies on population-based recombination and mutation, PSO on velocity-guided swarm dynamics, and ACO on pheromone-mediated probabilistic construction [9,11,23]. ACO variants have been applied specifically to pickup-and-delivery routing with time windows and heterogeneous fleets, which is structurally close to the problem addressed here [12]. These methods offer favorable trade-offs between solution quality and computational cost on medium-scale instances, but their performance is sensitive to parameter tuning and they must be re-run from scratch whenever demand or constraints change. Recent studies have accordingly investigated rich, hybrid, and adaptive routing strategies for heterogeneous fleet management and large-scale delivery optimization [10]. Despite these advances, metaheuristics remain fundamentally reactive: they optimize a static snapshot of the problem and do not learn transferable policies across instances.

2.3. Agentic AI and Intelligent Logistics Systems

Recent logistics research has increasingly explored intelligent coordination and adaptive decision-making mechanisms that move beyond static optimization. Blockchain-enabled architectures have been proposed to improve delivery traceability and decentralized coordination in logistics systems [24]. Sustainability-oriented routing approaches integrating CO_2-aware optimization and adaptive delivery mechanisms have emerged as an important research direction [25], alongside planning models for mixed fleets of electric and conventional vehicles in urban freight [26]. Reference architectures integrating autonomous delivery vans and sidewalk robots within a unified platform have been proposed to coordinate autonomous fleets and stakeholders in urban last-mile fulfillment [27], and exact routing formulations have begun to quantify the cost structure of coordinating trucks with autonomous delivery robots [19]. Digital-twin architectures coupling simulation with reinforcement learning have also been proposed as a coordination substrate for supply-chain and logistics operations [28].

In parallel, RL and multi-agent systems have introduced genuinely adaptive routing strategies for dynamic logistics environments, learning policies that generalize across instances rather than optimizing a single static problem [13,16,29]; recent surveys map the resulting design space for machine-learning-based vehicle routing [17]. Multi-agent formulations are particularly attractive for last-mile delivery because they model each vehicle as an autonomous decision maker that cooperates with the rest of the fleet. In this

context, Multi-Agent Proximal Policy Optimization (MAPPO)—an actor–critic method with a centralized critic and decentralized actors—and Multi-Agent Deep Q-Networks (MADQN)—a value-based approach with decentralized action-value estimation—have shown promising results for cooperative routing and decentralized transportation decision-making [14,15]. Related work has also combined deep RL with metaheuristics to solve multi-depot and dynamic routing problems, underscoring the complementarity of the two paradigms [30]. A recurring methodological caveat in this literature is that reported RL performance is highly sensitive to random seeds, implementation details, and evaluation protocol, so single-run comparisons must be interpreted with corresponding caution [31].

2.4. Research Gap

A qualitative comparison of representative approaches is summarized in Table 1. Reading across the columns, the gap this study targets can be stated precisely, in three parts.

Structural gap. Most studies address forecasting, fleet allocation, and routing as isolated tasks. Where integration is attempted it is usually unidirectional: a forecast is produced, frozen, and handed to an optimizer. The consequences of that interface—how forecast uncertainty should size a fleet, and how a fleet mix constrains what a router can achieve—are therefore rarely observable within a single study.

Comparative gap. Few studies place classical metaheuristics and agentic multi-agent RL inside the same architecture under identical, forecast-driven conditions and a single shared objective. Comparisons that exist typically differ simultaneously in instance family, objective, and search budget, so the reported ordering confounds the paradigm with its experimental treatment, leaving practitioners without guidance on when each should be preferred.

Evaluation gap. Selection layers are frequently proposed but seldom evaluated as decision rules in their own right. A selector is useful only if it beats both the best fixed solver and a trivial fallback, which requires instance-level regret against an oracle. This study closes the structural gap by construction and the comparative gap by holding instance, objective, and interface fixed across five solvers. It does *not* close the evaluation gap (Sections 1.2 and 5.5); making that distinction explicit is itself part of the contribution, since it defines what remains to be demonstrated.

Table 1. Comparison of representative studies across forecasting, routing, orchestration, and evaluation dimensions. "Orchestr." denotes an explicit mechanism that selects among alternative optimizers at run time; "Agentic RL" denotes learned multi-agent or deep-RL routing policies.

Study	Forecast	Routing	Orchestr.	Agentic RL	Focus
Mamede et al. [4]	✓	–	–	–	Deep learning demand forecasting
Xia et al. [5]	✓	–	–	–	Cross-city transfer forecasting
Chen et al. [9]	–	✓	–	–	Improved GA/ACO for VRPTW
Ky Phuc and Phuong Thao [12]	–	✓	–	–	ACO, heterogeneous fleet, time windows
Gao et al. [11]	–	✓	–	–	Hybrid ACO for CVRP
Pan and Liu [16]	–	✓	–	✓	DRL for dynamic/uncertain VRP
Arishi and Krishnan [30]	–	✓	–	✓	Multi-agent DRL, multi-depot VRP
Bogyrbayeva et al. [17]	–	✓	–	✓	Survey of ML methods for VRP
Guerreiro Augusto et al. [27]	–	✓	✓	–	Autonomous fleet reference architecture
Abideen et al. [28]	–	–	✓	✓	Digital twin with RL for logistics
This study	✓	✓	✓	✓	Unified forecast-driven pipeline; five solvers, one objective; orchestration specified, not yet measured

3. Materials and Methods

3.1. System Overview

The proposed framework is an orchestration architecture integrating demand forecasting, fleet allocation, distribution management, and route optimization within a unified last-mile delivery pipeline. As illustrated in Figure 1, the system combines forecasting and routing stages through a selection layer that chooses an optimization strategy according to operational conditions and instance characteristics.

The forecasting stage transforms delivery datasets into hourly zone-level demand sequences and applies AutoML-based forecasting using recurrent architectures (LSTM, GRU, BiLSTM, BiGRU, attention-enhanced LSTM), adapted across heterogeneous zones by transfer learning and automated hyperparameter optimization [5,6]. The routing stage converts forecasts into geographically constrained clusters, performs fleet allocation, generates VRPTW instances, and solves them with classical metaheuristics or agentic MARL solvers dispatched by the selection layer.

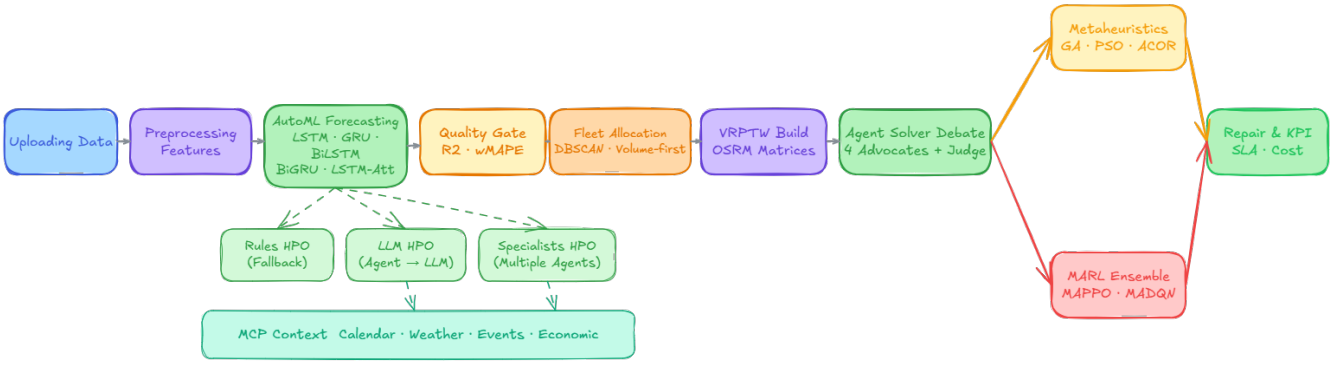


Figure 1. AI-driven orchestration framework for demand forecasting, fleet allocation, distribution management, and adaptive route optimization. Data flows left to right: raw delivery records are preprocessed into hourly zone-level series, forecast by the AutoML layer under a quality gate on R^2 and wMAPE, converted into clustered VRPTW instances with an allocated heterogeneous fleet and OSRM travel matrices, dispatched by the solver-selection stage to the metaheuristic or MARL family, and finally passed through coverage repair and KPI computation. The lower branch shows the four hyperparameter-orchestration modes of Section 3. Note that the context module exposes connectors for weather, event, and economic covariates in addition to the calendar; only the calendar covariates are used in the reported configuration, and no weather, event, traffic, or economic data enters any result in this paper.

3.2. Data Provenance

Because the interpretation of every result in this paper depends on what the underlying corpus actually is, this subsection states the provenance in full and separates what is real, what is transferred, and what is synthetic.

3.2.1. Source Corpus

The study uses LaDe (Last-mile Delivery), the industrial-scale last-mile delivery dataset publicly released by Cainiao, which provides courier trajectories, package-level records, and delivery timestamps across several Chinese cities [32]. Specifically, we use the single Hangzhou delivery partition, comprising 1,861,600 package records, 1392 couriers, and 17,714 areas of interest (AOIs) across 16 AOI classes. This is the only empirical delivery corpus used in this study.

3.2.2. Generation of the Casablanca Corpus

The Casablanca corpus is synthetic: it is generated from the LaDe Hangzhou partition rather than collected from Moroccan operations, and is constructed so as to reproduce the statistical structure of real last-mile demand within the Casablanca boundary. The generator is deterministic and seeded, and its parameters are listed in Table 2.

Table 2. Provenance of the Casablanca corpus by field group. "LaDe-calibrated" means generated from the LaDe Hangzhou partition under a structure-preserving transformation; "synthetic" means generated independently by this study's generator; "real" means taken directly from an external source.

Field Group	Status	Origin and Transformation
Corpus size	LaDe-calibrated	1,861,600 records, one generated per source record, preserving the source workload profile
Courier, AOI identifiers	LaDe-calibrated	1392 couriers, 17,714 AOIs; identifiers permuted under a fixed seed
Delivery coordinates	LaDe-calibrated	Per-axis CDF transform onto OSM relation 4072985 preserving quantile structure, mixed 0.25 with a uniform in-polygon component; land and coastline guards applied
Timestamps	LaDe-calibrated	Calendar re-based to 1 May–31 October 2026; time-of-day and weekday periodicity preserved
Parcel volume (m^3)	Synthetic	Log-normal $\mu = -3.51$, $\sigma = 0.70$, clipped $[0.002, 1.5]$; realized median 0.0299, mean 0.0382, p95 0.0946
Calendar covariates	Real (Moroccan)	Moroccan public holidays, Ramadan/Eid, school breaks, Friday and payday windows computed for the re-based dates
Road distances, durations	Real	OSRM `driving` profile over the OpenStreetMap Casablanca network

Spatial generation proceeds by a per-axis cumulative-distribution transform that maps the source demand density onto the Casablanca prefecture boundary (OpenStreetMap relation 4072985, simplified to approximately 500 m tolerance) while *preserving its quantile structure*, so that the relative concentration of demand between dense central districts and sparse peripheral ones is retained rather than replaced by a uniform draw. A uniform in-polygon component is mixed in at weight 0.25 so that the result is not a rigid re-scaling of the source density, and polygon and coastline guards constrain every generated point to land inside the boundary. Timestamps are re-based onto a six-month window, 1 May to 31 October 2026, preserving time-of-day and weekday periodicity, and courier and AOI identifiers are re-mapped. One synthetic record is generated per source record, giving a corpus of 1,861,600 records.

The corpus therefore inherits the correlation structure of real delivery operations—intraday and weekly demand rhythms, per-courier workload sequencing, AOI-level clustering, and the spatial concentration profile—while its geography is Casablanca's road network and administrative boundary. Because it is calibrated on a different city, it does not reproduce Casablanca's own demand distribution, which is not publicly documented. We accordingly frame the forecasting experiments as measuring adaptation under a mild, spatially induced distribution shift with the temporal signal preserved, rather than as validated transfer between two observed cities, and results should be read at that strength.

3.2.3. Synthetic Components

Per-parcel volume is not recorded in LaDe and is generated independently, from the log-normal distribution given in Table 2; it enters only the fleet-sizing stage, through the vehicle volume constraint. The calendar covariates, by contrast, are genuinely Moroccan: public holidays, Ramadan and Eid windows, school breaks, the Friday midday effect, and

the end-of-month payday window are computed for Morocco and joined onto the re-based timestamps. The corpus is therefore a deliberate hybrid—Moroccan calendar semantics over LaDe-calibrated demand geometry—so that calendar-driven structure the models exploit reflects the target market rather than the source city.

Augmentation is used only to diversify operating conditions in the forecasting stage. No reported metric is generated by augmentation; all metrics in Section 4 are computed against held-out observations of the generated corpus.

3.2.4. Aggregation, Coverage, and Splits

For the forecasting stage, records are aggregated into hourly, zone-level demand series, where $y_{z,t}$ is the number of delivery requests in zone z during hour t. The reported coverage of 99.4% is defined explicitly as

$$\text{coverage} = 100 \times \frac{|\{(z,t) : \text{an aggregated observation exists}\}|}{n_{\text{days}} \times 24 \times n_{\text{zones}}}, \quad (1)$$

that is, the proportion of cells of the complete zone-hour grid, spanning the observation window, for which an aggregated record exists. It is a measure of grid completeness, not of data quality or of agreement with any external ground truth. The denominator is the full Cartesian grid over the six-month window and the set of active zones.

All splits are chronological: the model is fitted on the earlier portion of the window and evaluated on a strictly later held-out period, with no shuffling across the boundary. Table 3 lists the input features by category. Two branches are used: a sequence branch consumed by the recurrent encoder, and a calendar-context branch consumed by a separate multilayer perceptron whose output is concatenated before the forecasting head.

Table 3. Input feature categories of the forecasting model. The lookback window is $L = 168$ h (one week) and the forecast horizon is 24 h. "(norm.)" denotes per-zone normalization applied before training.

Category	Branch	Features
Endogenous demand	Sequence	Normalized hourly demand $y_{z,t}$ (norm.)
Lagged demand	Sequence	Demand at $t-24$, $t-168$, $t-336$ h (norm.)
Rolling statistics	Sequence	24 h rolling mean and standard deviation, 168 h rolling mean (norm.), each shifted by one hour to prevent leakage
Cyclical time	Sequence	sin/cos encodings of hour-of-day, day-of-week, and month; weekend indicator
Calendar context	Calendar MLP	Public-holiday, holiday-eve and holiday-adjacent flags; normalized hours to next and since previous holiday; Ramadan flag and normalized Ramadan day; Eid window; school break; month-start and month-end flags
Calendar interactions	Calendar MLP	Friday flag, payday-window flag, and the products Ramadan × hour, Friday × hour, payday × day-of-month, holiday × hour
Zone identity	Both	Zone (cluster) identifier used to group series and to scope normalization

No exogenous traffic, weather, promotional, or price data is used. The model is therefore a calendar-conditioned autoregressive demand model; this is a deliberate scoping choice and also a limitation, since demand surges driven by unobserved promotions cannot be anticipated from these inputs.

3.3. Data Processing and AutoML Supervisor Module

For each zone, the system constructs supervised learning windows from the features of Table 3, producing an input tensor $X_{z,t-L+1:t}$ used for short-term demand prediction. To adapt the forecasting layer across heterogeneous zones, the framework integrates an AutoML supervisor responsible for data diagnosis, model selection, and hyperparameter orchestration. Automated model selection and hyperparameter optimization of this kind are established components of AutoML forecasting systems [6].

A diagnostic score

$$S_z = f(h_z, \rho_z, \sigma_z, \delta_z, \kappa_z) \tag{2}$$

is computed for each zone over the observed series $\{y_{z,t}\}_{t=1}^{T_z}$, where h_z is history length; $\rho_z = T_z^{-1} \sum_t \mathbb{1}[y_{z,t} = 0]$ the zero-demand ratio, quantifying intermittency; σ_z the seasonality strength, estimated from dominant spectral and autocorrelation peaks at daily and weekly lags; $\delta_z = |\mu_z^{(2)} - \mu_z^{(1)}| / (\mu_z^{(1)} + \varepsilon)$ the drift score, capturing non-stationarity as the normalized difference in mean demand between the first and second halves of the series; and κ_z the calendar density, computed as in Equation (1) restricted to zone z. Zones with insufficient history or excessive intermittency are flagged non-forecastable and handled by a conservative fallback estimator—the zone's historical hour-of-week means—rather than by deep models. For forecastable zones, the diagnostic score guides both backbone selection from the model registry and hyperparameter orchestration.

Hyperparameter optimization supports four orchestration modes: deterministic rule-based configuration, single-agent optimization, multi-specialist debate, and an agentic loop with validation, retry, and fallback. All generated configurations are validated against operational and training bounds before execution; any violating configuration is rejected and replaced by the deterministic one. The contribution of each of these mechanisms has not been isolated by ablation in this study (Limitation L4).

3.4. Transfer Learning Forecasting Architecture

The forecasting stage relies on transfer-learning-enhanced recurrent architectures. Pretrained representations are initialized on the data-rich source partition and fine-tuned on target zones [5]. For each forecastable zone, $X_{z,t-L+1:t}$ is processed through recurrent encoder layers capturing temporal dependencies and seasonal patterns, while the calendar-context branch is encoded separately and concatenated; attention-enhanced variants additionally weight relevant historical observations. The resulting representation H_t is projected into multi-horizon forecasts $\hat{Y}_{z,t+h} = g_\phi(H_t)$ by the forecasting head $g_\phi(\cdot)$. Training uses adaptive learning-rate scheduling, automated hyperparameter optimization, and validation-based selection with early stopping.

Performance is evaluated over N evaluation points using

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|, \quad \mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}, \quad \mathrm{wMAPE} = \frac{\sum_i |y_i - \hat{y}_i|}{\sum_i |y_i|}, \tag{3}$$

together with $R^2 = 1 - \sum_i (y_i - \hat{y}_i)^2 / \sum_i (y_i - \bar{y})^2$, where y_i and $\hat{y}_i$ are observed and predicted demand and $\bar{y}$ the mean observation. MAE and RMSE express error in delivery-count units, wMAPE is scale-independent and stable under intermittent demand, and R^2 quantifies explained variance.

Upper-Bound Forecast and Its Calibration

Because fleet sizing is asymmetric in cost—under-provisioning fails deliveries, while over-provisioning wastes a shift—the forecasting head emits, in addition to the point

forecast $\hat{Y}_{z,t+h}$, an upper-bound estimate $\hat{U}_{z,t+h}$ intended to be exceeded only rarely. The bound is calibrated post hoc on held-out data by finding the smallest multiplicative scale s such that the empirical coverage attains a target quantile q:

$$s^\star = \min\left\{ s \in [0.1,\ 3.0] \ :\ \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[y_i \le s\,\hat{U}_i\right] \ge q \right\}, \qquad q = 0.85, \tag{4}$$

solved by bisection over s and capped at $s = 3$ to prevent degenerate bounds. The calibrated bound $s^\star\hat{U}$ is what the fleet-allocation stage consumes. This defines and calibrates the bound; it does not establish its operational value. A comparison of point-based against upper-bound-based planning on fleet size, service level, and cost was not run in this study and is stated as an open evaluation in Section 5.5.

3.5. Fleet Allocation

Forecast outputs are spatially aggregated and clustered into geographically coherent delivery regions. For each cluster, the framework estimates the required fleet from the calibrated upper-bound demand, a safety factor of 1.15, and per-type capacity. Allocation respects the order-capacity constraint

$$\sum_{i \in R_k} d_i \le C_k \tag{5}$$

where R_k is the set of requests assigned to vehicle k, d_i the demand at node i, and C_k the capacity of vehicle k; a parallel constraint applies to loaded volume using the synthetic per-parcel volumes. This stage produces a heterogeneous fleet mix—counts of e-bikes, cargo bikes, and vans—which is passed to the routing stage as a fixed roster.

3.6. VRPTW Instance Generation

The routing graph is $G = (V, E)$ with $V = \{0\} \cup V_c$, where node 0 is the depot and V_c the customer set. Travel distances c_{ij} and durations t_{ij} come from OSRM over the real Casablanca road geometry rather than Euclidean approximations. Letting K be the vehicle set and $x_{ijk} = 1$ if vehicle k traverses arc (i, j), the canonical objective minimizes travel cost,

$$\min \sum_{k \in K} \sum_{(i,j) \in E} c_{ij}\, x_{ijk}, \tag{6}$$

subject to each customer being visited exactly once, capacity as in Equation (5), and service beginning within the node's window,

$$a_i \le \omega_i \le b_i, \qquad \forall\, i \in V_c, \tag{7}$$

where ω_i is the service start time at node i, $[a_i, b_i]$ its window, and $\omega_j \ge \omega_i + s_i + t_{ij}$ enforces temporal consistency (s_i being the service duration).

Two properties of this formulation govern how the results should be read. First, the time windows are *soft*: violating $[a_i, b_i]$ is permitted and priced by a penalty rather than forbidden, the only hard temporal constraint being the operating day $[\,08{:}00,\ 22{:}00\,]$. Second, all vehicle types share a single OSRM `driving` distance matrix, per-type durations being obtained by dividing distances by cruise speed; bicycle infrastructure, pedestrian zones, and vehicle-specific access restrictions are therefore *not* modeled, and the reported advantage of light vehicles reflects cost and capacity only, not differential network access. Table 4 gives the complete specification.

Table 4. Complete specification of the VRPTW benchmark instance and the generator rules that produce it. Monetary values are in Moroccan dirham (MAD) per shift.

Component	Value	Definition/Rule
Customers $\lvert V_c \rvert$	50	Sampled from the forecast cluster's candidate point pool for the selected hour
Depots	1	Located at $(33.59364,\ -7.55844)$; window $[08{:}00, 22{:}00]$; zero service time
Demand per node	1 order	Each customer node is one delivery request
Time window	$\pm$60 min	Centered on the forecast hour h: $[h - 60$ min, $h + 60$ min$]$, clipped to the operating day
Window stagger	$\pm$30 min	Offset by normalized haversine depot distance: nearest customers shift earlier, farthest later; minimum width 10 min
Operating day	08:00–22:00	Hard constraint; the only hard temporal bound
Lateness penalty	0.5 per second	Applied to arrival outside the soft window
Distance/duration	OSRM `driving`	Single $(N+1) \times (N+1)$ table call on the OpenStreetMap Casablanca extract; per-type durations derived by cruise speed
E-bike	cap. 30	20 km/h; service 210 s; shift 8 h; utilization 0.85; 80 MAD; 0.00 kg CO_2/km
Cargo bike	cap. 50	15 km/h; service 240 s; shift 8 h; utilization 0.80; 60 MAD; 0.06 kg CO_2/km
Van	cap. 120	35 km/h; service 300 s; shift 8 h; utilization 0.75; 200 MAD; 0.21 kg CO_2/km

3.7. Routing Optimization

The metaheuristic layer integrates GA, PSO, and ACO for population-based and probabilistic route exploration [9,11,23]. All three share a $2n$-dimensional random-key encoding in which the first n keys determine the customer visit order and the remaining n keys determine vehicle assignment, making the search directly fleet-aware. In parallel, the framework evaluates agentic routing systems based on MAPPO and MADQN, in which each vehicle is an autonomous routing agent [14–16]. MAPPO trains a shared decentralized actor with a centralized critic conditioned on the joint fleet state; MADQN follows a value-decomposition design with experience replay and target networks. Execution is decentralized in both cases.

3.7.1. Shared Fitness Function

Rather than optimizing travel distance alone, all five solvers minimize one shared fitness that augments Equation (6) with operational penalties:

$$F(\sigma) = D(\sigma) + \lambda_{\text{tw}} P_{\text{tw}}(\sigma) + \lambda_u P_u(\sigma) + \lambda_s P_s(\sigma) + \lambda_k K(\sigma) + \lambda_c C(\sigma) + \lambda_b B(\sigma), \quad (8)$$

where σ is a candidate solution, D is total travel distance in kilometres, P_{tw} is cumulative lateness in seconds, P_u is the count of unserved customers, $P_s = \max(0,\ \tau - \text{SLA}(\sigma))$ is the shortfall in percentage points below the service-level target τ, K is the number of vehicles used, C is total fleet shift cost, and B is workload imbalance measured as the summed absolute deviation of per-vehicle delivery counts from their mean. The weights actually used are given in Table 5; they are reported in full because the relative magnitudes, not the functional form, determine solver behavior.

Table 5. Weights of the shared fitness function, Equation (8). Distance is measured in kilometres, so every weight is expressed in kilometre-equivalent cost units.

Term	Weight	Interpretation
Distance D	1.0	One cost unit per kilometre travelled
Lateness P_{tw}	$\lambda_{\text{tw}} = 0.5$	Per second of arrival outside the soft window
Unserved P_u	$\lambda_u = 10^6$	Effectively a hard constraint on full coverage
SLA shortfall P_s	$\lambda_s = 50$	Per percentage point below the target $\tau = 80\%$
Vehicle count K	$\lambda_k = 5.0$	Kilometre-equivalent cost of activating one more vehicle
Fleet cost C	$\lambda_c = 1/200$	Shift cost normalized so that one van equals one cost unit
Imbalance B	$\lambda_b = 0.5$	Per delivery of deviation from the mean vehicle load

The 10^6 weight on unserved demand makes full coverage effectively mandatory, and this choice interacts with the repair mechanism described below in a way that materially shapes the results. For the RL solvers, the per-step reward is the negative increment of this fitness, $r_t = -\Delta F_t$, so maximizing expected return is equivalent to minimizing $F(\sigma)$.

3.7.2. Action Masking and Coverage Repair

At each decision step an agent observes the operational state

$$s_t = (v_t, c_t, \tau_t, d_t) \tag{9}$$

where v_t is the current vehicle position, c_t the remaining capacity, τ_t the remaining time budget, and d_t the unserved demand, together with the $K = 20$ nearest unvisited customers and the acting vehicle's type features. The agent selects the next node to visit or a return-to-depot action.

Two implementation details must be reported precisely, because together they explain the routing results of Section 4.

Masking is capacity-based, not time-window-based. The action mask admits the depot at all times and admits customer i if and only if i is unvisited and the acting vehicle's load plus d_i does not exceed its capacity. Time windows are *not* consulted when constructing the mask. This is consistent with the soft-window formulation—a late arrival is a priced outcome, not an illegal one—but it means that masking cannot and does not prevent time-window violations. Furthermore, the mask is evaluated one step at a time, so even a capacity-legal action may commit a vehicle to a trajectory whose later stops are unreachable in time.

Coverage is enforced post hoc by repair. Because unserved demand carries the 10^6 penalty, every returned solution passes through a cheapest-insertion repair that assigns any remaining customer to the position of minimum added distance. The repair checks time feasibility in the general case, but with three documented relaxations: routes shorter than five stops skip the feasibility test and rely on downstream decoding to price the violation; routes longer than twenty stops restrict the insertion scan to positions near the geometrically closest existing stop; and when an explicit vehicle-count cap has been reached, insertion proceeds on distance alone, ignoring timing, so that the customer is served late rather than dropped.

The composition of these two mechanisms is the key to interpreting agentic results: a policy is free to construct short routes that defer time-constrained deliveries, and repair will then complete coverage without restoring feasibility. The resulting solution can therefore exhibit a short total distance and a low service level simultaneously. We report the service level of every solution alongside distance for this reason, and we do not treat distance as a standalone efficiency measure.

3.8. Solver Selection Layer

The selection layer treats the five solvers as interchangeable strategies behind one interface: each accepts an instance and a fleet roster, and returns a solution scored by Equation (8). Normatively, the layer should implement

$$\pi^\star = \arg\max_{\pi \in \Pi} U(\pi \mid \Phi), \qquad \Pi = \{\mathrm{GA}, \mathrm{PSO}, \mathrm{ACO}, \mathrm{MAPPO}, \mathrm{MADQN}\}, \tag{10}$$

where Φ is the instance context descriptor and U trades expected solution quality against computational budget. We state clearly that Equation (10) is a specification, not a fitted estimator. The deployed system does not estimate U from data; no regression, ranking model, or calibration set exists for it. What is implemented instead is the following procedure, whose position within the routing stage is shown in Figure 2.

1. Descriptor. Compute Φ: customer count, spatial extent and depot-distance distribution, mean and dispersion of window width, fleet composition, and an OSRM probe of real travel time from depot to a sample customer.
2. Advocacy. Four advocate agents—GA, ACO, PSO, MARL—receive Φ and return a structured argument with self-reported confidence in $[0, 100]$, with access to OSRM tools for travel data.
3. Adjudication. A judge receives Φ, all four arguments, and the probe, and returns the primary metaheuristic, whether to additionally run a MARL solver and which algorithm, an execution mode in {classical only, MARL only, both}, and per-solver hyperparameters.
4. Validation and capping. Returned hyperparameters are clamped to the admissible ranges of Table 6; out-of-range or malformed responses are discarded.
5. Fallbacks. Absent a language-model backend, or on failed validation, the layer reverts to a fixed configuration (ACO primary, MAPPO enabled, both modes). If the selected strategy returns no solution within its time budget, or one with unserved demand, it falls back to the next-ranked strategy.

Two consequences follow. Selection quality is bounded by the judge's reasoning over a textual descriptor; it is not learned from outcome data and carries no optimality guarantee. And because the fallback fixes ACO as default, the layer degenerates to a fixed-solver policy without a language-model backend. Whether the full procedure beats that degenerate policy—or an oracle picking the best solver per instance—requires a multi-instance evaluation and is not answered here.

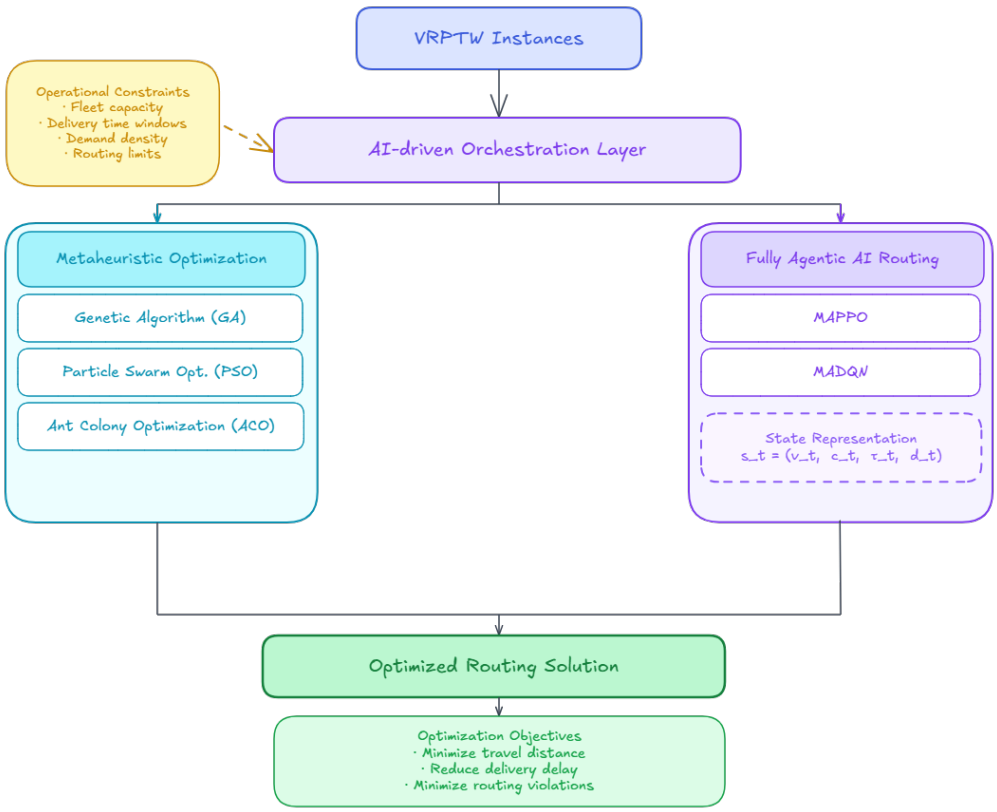


Figure 2. Routing layer of the framework. VRPTW instances enter the selection layer, which dispatches to either the metaheuristic family (GA, PSO, ACO) or the agentic MARL family (MAPPO, MADQN) under the operational constraints shown at left; both branches return a solution scored by the shared fitness of Equation (8), and both consume the same state representation $s_t = (v_t, c_t, \tau_t, d_t)$ defined in Section 3.7.2. The selection procedure itself—instance descriptor, advocacy, adjudication, hyperparameter capping, and the deterministic and feasibility fallbacks—is specified step by step in Section 3.8 rather than in this diagram.

3.9. Computational Budgets and Experimental Protocol

Table 6 reports the configuration actually used to generate the reported solutions. The rightmost column makes the search budget explicit in a comparable unit—the number of candidate solutions evaluated—because these budgets are *not* equal across solvers.

Table 6. Configuration of the evaluated routing strategies. Metaheuristics use the *mealpy* library; MARL solvers use Stable-Baselines3/sb3-contrib. ACO uses the continuous-domain variant ACO_R; MADQN follows a value-decomposition (VDN) design with parameter sharing and action masking; MAPPO uses a maskable PPO policy with parameter sharing and centralized training. Both MARL solvers use a two-layer $[64, 64]$ multilayer perceptron. Budgets are given as population $\times$ iterations, i.e., the number of candidate solutions evaluated.

Strategy	Key Parameters	Setting and Search Budget
GA	population/generations/p_c/p_m	80/100/0.9/0.02—8000 evaluations
PSO	swarm/iterations/w/c_1, c_2	30/50/0.7/2.05, 2.05—1500 evaluations
ACO_R	ants/iterations/q/ζ	25/60/0.5/1.0—1500 evaluations
MAPPO	lr/γ/GAE λ/clip/entropy	3×10^{-4}/0.995/0.95/0.2/0.02—10^5 environment steps
MADQN	lr/γ/replay/batch/ϵ-decay	3×10^{-4}/0.995/5×10^4/64/1.0 $\rightarrow$ 0.02—10^5 environment steps

Budget asymmetry. GA is allotted 8000 candidate evaluations against 1500 each for PSO and ACO—a factor of 5.33. These are the hard caps imposed by the configuration layer of Section 3.8, and they were not equalized for the reported run. Any GA-versus-metaheuristic comparison in Section 4 is therefore confounded with budget, and we do not attribute GA's outcome to algorithmic superiority. The MARL budget, expressed in environment steps, is not commensurable with candidate evaluations at all, since training cost amortizes across instances while metaheuristic cost is incurred per instance.

Number of runs. Each solver was executed once, under a fixed random seed. No repeated-seed replication was performed, and consequently no dispersion statistics, confidence intervals, or significance tests are reported anywhere in this paper. This is substantive rather than presentational: single-run comparisons of stochastic optimizers, and of RL agents especially, are unreliable indicators of expected performance [31], and a defensible comparison would require repeated seeds analyzed by rank-based procedures across multiple instances [33].

Training and test separation. The MARL policies were trained on the same instance on which they are reported, using curriculum learning and a behavior-cloning warm start from a greedy constructive heuristic. There is thus no train/test separation for the agentic solvers, and their numbers are fitted-instance performance, not generalization. The metaheuristics, performing no learning, are unaffected—which makes the comparison asymmetric in a second way beyond budget. Section 5.5 specifies the protocol required to remove both asymmetries.

Software. All experiments were run under Python 3.11.0. The forecasting backbones were implemented in TensorFlow 2.18.0 with Keras 3.6.0; hyperparameter search used Optuna 3.6.2, and preprocessing used scikit-learn 1.5.2, NumPy 1.26.0, and pandas 2.2.3. The metaheuristic solvers (GA, PSO, ACO_R) used mealpy 3.0.3. The agentic solvers used Stable-Baselines3 2.7.1 and sb3-contrib 2.7.1 on PyTorch 2.10.0 (CUDA 12.6), with environments built on Gymnasium 1.2.3. Road-network distances and travel times were obtained from a locally deployed Open Source Routing Machine (OSRM) 5.26.0 instance (Docker image `osrm/osrm-backend`), served by `osrm-routed` under the multi-level Dijkstra algorithm with the car profile, and built over the Geofabrik Morocco OpenStreetMap extract (https://download.geofabrik.de/africa/morocco-latest.osm.pbf,

accessed on 17 April 2026). Running OSRM locally fixes the road network for the duration of the study, so the travel-time matrices are reproducible from that extract.

4. Results

Forecasting performance was assessed on a chronologically held-out evaluation period. The routing strategies were compared on one shared benchmark instance, specified completely in Table 4, with each solver executed once under a fixed random seed. This protocol isolates the effect of the optimization strategy by holding demand, fleet, and constraints constant, but—as stated in Section 3.9—it does not control for search budget and does not support statistical inference. The results below are therefore reported as a single-instance case comparison, and no general ordering of solver paradigms is claimed from them.

4.1. Forecasting Performance

Table 7 summarizes performance across the five recurrent backbones. The bidirectional LSTM achieved the best values on every metric (MAE = 165.87, RMSE = 370.62, wMAPE = 29.4%, $R^2 = 0.865$).

Table 7. Forecasting performance on the generated Casablanca corpus (1,861,600 delivery records; zone-hour grid coverage 99.4% as defined in Equation (1); six-month window, 1 May–31 October 2026). Chronological hold-out, single training run per backbone. Best value in each column in bold.

Model	MAE	RMSE	wMAPE (%)	R^2
LSTM	172.45	385.43	30.6	0.854
GRU	170.82	384.73	30.3	0.855
BiLSTM	**165.87**	**370.62**	**29.4**	**0.865**
BiGRU	176.02	396.90	31.2	0.845
Attention-LSTM	173.40	388.70	30.8	0.851

The substantive observation is the narrowness of the spread rather than the identity of the winner. All five backbones fall within 0.020 of R^2 and 1.8 percentage points of wMAPE, and bidirectional encoding confers no consistent advantage, BiLSTM ranking first and BiGRU last. Since each was trained once, the BiLSTM-to-next-best gap ($\Delta R^2 = 0.010$) is of the same order as the seed-to-seed variation typically observed for recurrent models on demand series, and we therefore do not claim BiLSTM is significantly better. What the table supports is the weaker conclusion that the choice among these backbones is not the decisive design decision—which is what motivates delegating it to a diagnostic-driven supervisor rather than hand-selecting per zone.

Three caveats bound this table, and together they are why RQ1 is answered at the level of "backbone choice is not decisive" rather than as demonstrated robustness. No naive or non-recurrent baseline is reported, so a wMAPE of 29.4% cannot be judged strong or weak because the intrinsic predictability of these series is not established; errors are aggregate, so zone-level distributions—where an operator's exposure actually lies—are invisible; and no ablation separates transfer learning, AutoML selection, the diagnostic rules, or the fallback estimators. Limitation L4 specifies the remedy.

4.2. Demand Clustering and Fleet Allocation

Forecast outputs were spatially aggregated into geographically coherent delivery clusters for VRPTW generation, balancing delivery density, coverage, and routing feasibility while preserving spatial proximity. Using the calibrated upper-bound demand of Section 3.4, a safety factor of 1.15, and the per-type capacities of Table 4, the framework produced a heterogeneous fleet mix for each cluster. Figure 3 shows a representative

operational snapshot: demand clustering, sub-zone generation, vehicle allocation, and fleet distribution over the Casablanca environment.

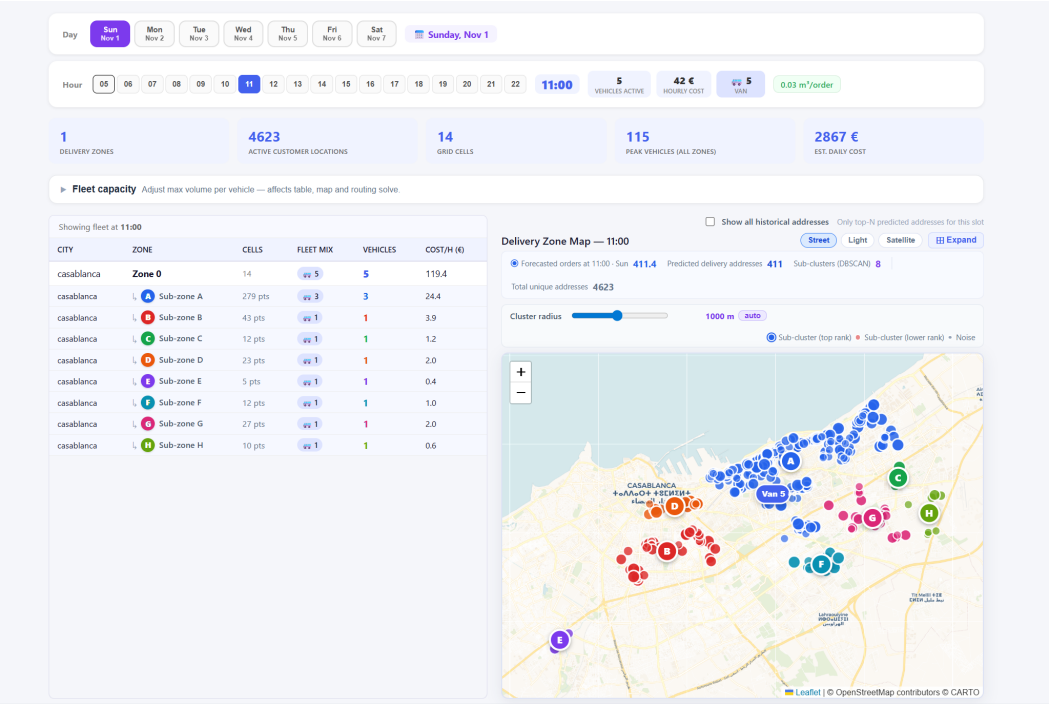


Figure 3. Demand clustering and heterogeneous fleet allocation generated by the orchestration framework within the Casablanca operational environment.

4.3. Worked Example: One Casablanca Planning Cycle

To make the pipeline concrete, we trace one planning cycle end to end, using the benchmark instance of Table 4.

Forecast. For the target cluster and delivery hour h, the AutoML supervisor computes the zone diagnostics, confirms the zone is forecastable, selects a backbone, and emits a point forecast $\hat{Y}$ and the calibrated upper bound $s^{\star}\hat{U}$.

Fleet. The upper bound, inflated by the 1.15 safety factor and divided by per-type effective capacity under both order and volume constraints, yields a vehicle roster. The roster is fixed before routing, which is what makes the comparison fleet-aware: solvers choose how many slots to activate and of which type, but cannot invent capacity.

Instance. Fifty delivery points are drawn from the cluster's candidate pool, each receiving a soft window of ± 60 min around h, staggered by up to ± 30 min by normalized distance from the depot at $(33.59364, -7.55844)$ so that near customers are scheduled earlier. A single OSRM `driving` table call returns the 51×51 matrices, from which per-type durations are derived by cruise speed.

Selection. The descriptor Φ—50 customers, mean window width 120 min, depot-distance profile, fleet mix—is passed to the stages of Section 3.8. For the reported experiment we *bypassed* selection and executed all five strategies on the same instance, so the comparison is not mediated by the selector.

Solve, repair, report. Each solver returns a solution, which passes through coverage repair (Section 3.7.2) before being scored by Equation (8) and returned with route geometry, per-vehicle loads, and service level. In the GA solution, five slots are activated—one e-bike, two cargo bikes, two vans—serving all 50 customers over 45.4 km with no window violated.

4.4. Routing Optimization Performance

Table 8 reports the five strategies on the shared instance, ordered feasibility-first: SLA-compliant solutions appear first, ranked by the shared fitness of Equation (8), with the infeasible solution last. This ordering is deliberate. Ranking by raw distance would place MAPPO first, which would be misleading, since its distance advantage is obtained by not serving customers on time.

Table 8. Routing performance on the shared 50-customer Casablanca VRPTW instance; one run per solver at a fixed seed. Fleet composition is e-bikes/cargo bikes/vans (E/C/V). Penalty is cumulative lateness in seconds. F' is the shared fitness of Equation (8) recomputed from the reported columns using the weights of Table 5, *excluding* the workload-imbalance term, which cannot be reconstructed because per-vehicle loads are not tabulated; F' is therefore a lower bound on each solution's true fitness. Rows are ordered feasibility-first, then by F'. Best feasible solution in bold.

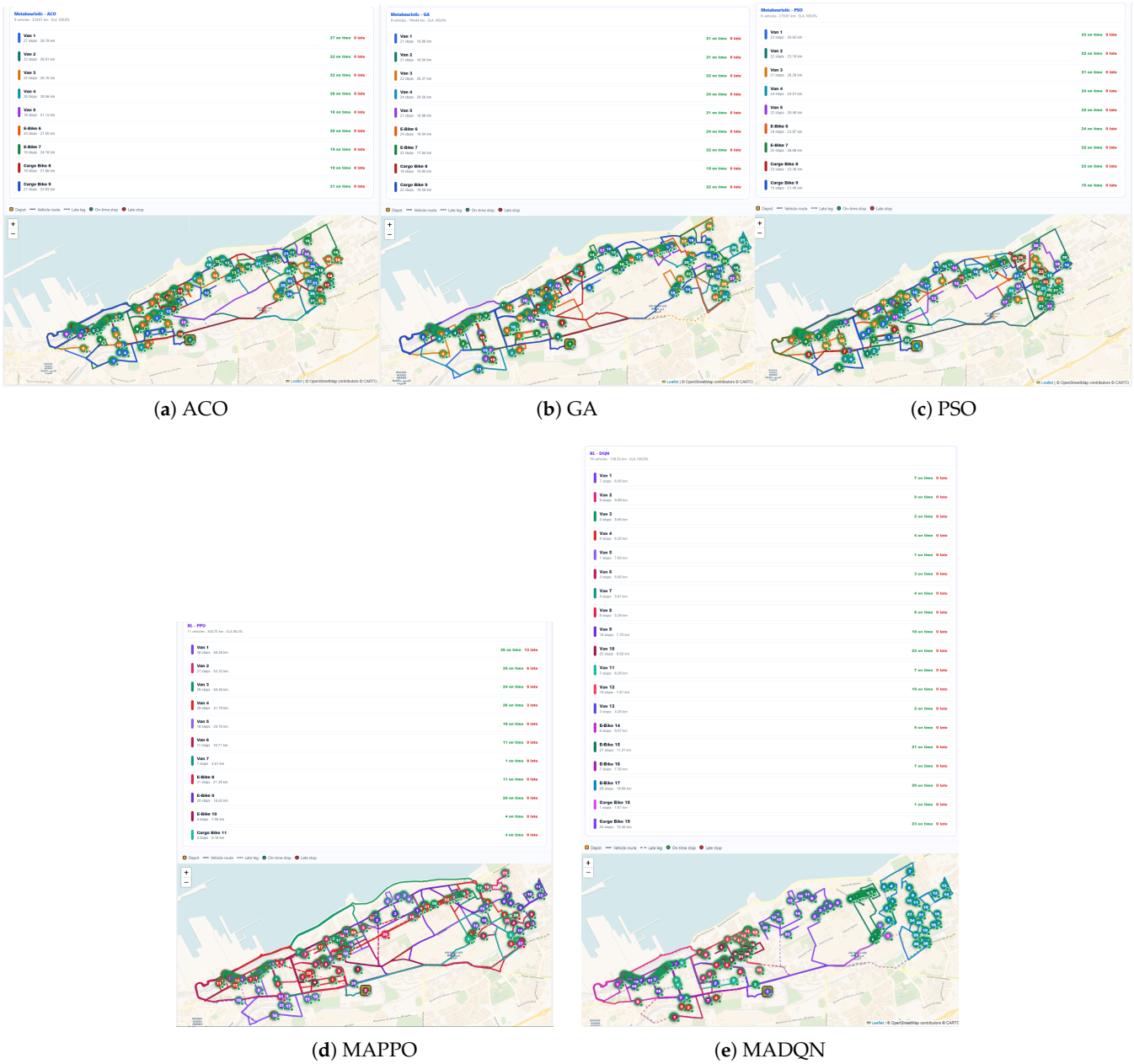
Method	Vehicles	Fleet (E/C/V)	Distance (km)	SLA (%)	Penalty (s)	F'
GA (metaheuristic)	**5**	**1/2/2**	**45.4**	**100.0**	**0**	**73.4**
ACO (metaheuristic)	7	2/1/4	54.7	100.0	0	94.8
PSO (metaheuristic)	7	2/1/4	56.3	100.0	0	96.4
MADQN (agentic)	8	2/2/4	69.6	100.0	0	115.0
MAPPO (agentic)	3	0/0/3	30.0	62.8	13,336	7576.0

Four of the five strategies returned fully compliant solutions. Among them GA was lowest-cost on every dimension—five vehicles, 45.4 km, lowest fitness—while ACO and PSO each needed seven vehicles over 54.7 km and 56.3 km, and MADQN was the most conservative at eight vehicles and 69.6 km.

GA's result must be discounted for budget. Per Section 3.9, GA evaluated 8000 candidate solutions against 1500 for PSO and ACO. A 5.33× advantage is a sufficient explanation for a gap of this size on a 50-customer instance, and nothing here separates the algorithm's contribution from the budget's. We therefore report GA as the best solution *found under this configuration* and decline to conclude that GA is the superior metaheuristic for this problem class.

MAPPO's 30.0 km is not an efficiency result. MAPPO produced the shortest distance using three vans, but served only 62.8% of deliveries within their windows, accumulating 13,336 s—roughly 3.7 h—of lateness. The mechanism is that of Section 3.7.2: the mask enforces capacity and non-repetition only, so the policy was never prevented from selecting a node it could not reach in time; having minimized distance and vehicle count, it left customers whose windows had effectively expired, and cheapest-insertion repair then completed coverage on distance grounds rather than timing grounds. The short route is thus the *signature* of the violation, not an achievement alongside it—the distance is low precisely because the constraints that lengthen compliant routes were not respected. Priced by the shared fitness, MAPPO scores $F' = 7576$ against GA's 73.4, roughly a hundredfold worse, despite travelling 34% fewer kilometres.

Two points bound the agentic rows. No customers were left unserved by any solver, since repair guarantees coverage by construction; the meaningful feasibility measure here is therefore service level, not unserved count. And both MARL policies were trained on this same instance, so their numbers describe fitted-instance behavior—the gap between MAPPO and MADQN, at identical budget, points to reward shaping and learning dynamics rather than an intrinsic ceiling on multi-agent routing. The route geometries underlying these rows are shown for all five solvers in Figure 4.



(**a**) ACO (**b**) GA (**c**) PSO

(**d**) MAPPO (**e**) MADQN

Figure 4. Representative routing visualizations produced by the evaluated strategies within the Casablanca VRPTW operational environment: (**a**) Ant Colony Optimization; (**b**) Genetic Algorithm; (**c**) Particle Swarm Optimization; (**d**) MAPPO cooperative routing; (**e**) MADQN agentic routing. These snapshots illustrate qualitative spatial routing behavior and are provided for visual comparison; they are not one-to-one renderings of the tabulated solutions. Quantitative performance is reported in Table 8.

4.5. Summary of Findings

Four findings are supported, each bounded by one corpus, one instance, and one run per method: (i) backbone choice is not the decisive forecasting decision, since five architectures span only 0.020 in R^2 and directionality does not order them consistently; (ii) constraint handling rather than paradigm separates the routing solvers, the two agentic solvers occupying opposite extremes of the feasibility spectrum while the three metaheuristics cluster as compliant; (iii) distance is not a valid standalone efficiency metric under soft windows, MAPPO's shortest route scoring roughly a hundredfold worse once lateness and

SLA shortfall are priced; and (iv) the metaheuristic ordering is confounded with search budget. These are developed in Section 5 and restated with their scope in Section 6.

Equally explicitly, the experiments do *not* establish that the orchestration layer improves end-to-end performance, that transfer learning or AutoML selection contributed measurably to accuracy, that upper-bound planning outperforms point planning, or that any solver ordering generalizes beyond this instance. Section 5.5 specifies what each would require.

5. Discussion

5.1. Answers to the Research Questions

RQ1. The forecasting layer attained R^2 between 0.845 and 0.865 and wMAPE between 29.4% and 31.2% across five recurrent backbones. The second half of the question is the more informative one: backbone choice accounts for at most 0.020 of R^2, and bidirectional encoding does not order the models consistently. For short-horizon zone-level demand of this kind, the recurrent cell is therefore not where design effort should be concentrated, which is a defensible argument for delegating that choice to a diagnostic-driven supervisor. We stop short of the stronger claim that the layer is *robust* or achieves accuracy *with minimal manual tuning*, which would require the baselines, ablations, and replication specified in L4.

RQ2. The paradigm label does not predict feasibility behavior, whereas the treatment of time windows does. The two agentic solvers, sharing an environment, a reward, and a training budget, landed at opposite ends of the feasibility spectrum, while the three metaheuristics clustered together as compliant. This is the central diagnostic result of the study, and Section 3.7.2 traces it to a mechanism rather than to a property of learning: because the mask enforces capacity but not timing, and coverage is restored post hoc by distance-driven insertion, a distance-minimizing policy can produce a short, well-covered, badly late solution that looks efficient under any distance-only metric. The quantitative ordering in Table 8 is far weaker, resting on one instance, one seed, and unequal budgets; we report it as a case observation, not as evidence about the relative merit of GA, PSO, and ACO.

RQ3. The selection layer is specified, implemented, and integrated: five solvers are interchangeable behind one descriptor, one objective, and one fallback contract (Section 3.8). That is an answer at the level of mechanism, not of measured benefit, and we do not present it as one. Equation (10) names a utility function the deployed system does not estimate; there is no calibration set, no fitted ranking model, and no reported selection regret. The claim that adaptive selection outperforms committing to a single solver therefore remains a hypothesis rather than a result. We regard stating this plainly as more useful than presenting the architecture's existence as its own validation; L3 defines the experiment that would settle it.

5.2. Interpretation

The forecasting and routing findings converge on one methodological point: in an integrated pipeline, the interfaces between stages matter more than the choice of component within a stage. Swapping BiLSTM for GRU moves R^2 by 0.010; leaving time windows out of an action mask moves service compliance by 37 percentage points. The stages of a last-mile pipeline are not independently optimizable modules whose local metrics compose into global quality, because each stage's metric is defined over quantities the next stage does not consume in the same form. A forecast is scored by conditional-mean error but consumed as a capacity bound; a route is scored by distance but consumed as a service promise.

This also explains why the efficiency–feasibility contrast reported here is not the trade-off it initially appears to be. A genuine trade-off would mean shorter routes must be paid

for in service level; instead, GA served every customer on time in 45.4 km, so 30.0 km at 62.8% compliance is not a different point on an efficient frontier but a dominated solution once its violations are priced. Reporting it as a distance win would misinform an operator. These observations are consistent with studies reporting that RL routing can approach or surpass classical methods on raw efficiency while requiring careful constraint handling to guarantee feasibility [16,17,30], and they reinforce the value of feasibility-first reporting.

5.3. Implications

Theoretical implications. The study is a concrete counterexample to the assumption that solver comparisons can be read off objective values without inspecting constraint semantics. Where time windows are modeled softly—common in practice, since operators do accept late deliveries at a cost—objective and constraint set are entangled, and a learned policy will exploit whatever the mask fails to forbid. Feasibility should therefore be a reported dimension rather than a precondition in RL routing research, and step-wise masking, myopic by construction, is an incomplete substitute for constraint enforcement over complete routes. For orchestration research the parallel point is that a selector not evaluated against an oracle and a best-fixed-solver baseline cannot be distinguished from a constant policy, so those baselines should be mandatory.

Managerial implications. Three follow from what was measured. Effort spent choosing among comparable forecasting architectures is likely less productive than effort spent calibrating the upper bound that drives fleet sizing, since the former moved accuracy by 0.020 of R^2 while the latter sets how many shifts are paid for. Before adopting a learning-based router, an operator should demand service-level reporting alongside distance and verify how the system treats deliveries it cannot complete on time, since a policy evaluated on distance alone can degrade contracted service while appearing to improve. And the fallback contract is itself an adoption enabler: because the architecture reverts to a deterministic metaheuristic when a learned policy is untrained or infeasible, SLA-critical decisions need not rest on a black-box policy—though the cost of that assurance is unquantified here.

Social implications. The fleet formulation prices e-bikes and cargo bikes below vans, so cost minimization shifts volume toward light, low-emission vehicles where capacity and timing permit, reducing local emissions and noise in dense residential zones. Two cautions belong alongside that. The model does not represent bicycle infrastructure or access restrictions, so it may recommend light-vehicle assignments that are unsafe on streets lacking protected lanes. And courier working conditions—shift length, route density, and the pressure created by tight windows—are essentially unmodeled, workload balance entering only as a mild penalty. An optimizer tuned purely on cost and distance can degrade those conditions while improving its reported metrics, which is the same failure mode, in another dimension, as the distance-versus-service effect of Section 4.4.

5.4. Computational Considerations

The complexity profiles of the two solver families differ in kind. Metaheuristics incur no training cost but must be re-run per instance, so cost scales with planning cycles. Learning-based routers invert this, requiring substantial offline training—here 10^5 environment steps with curriculum learning and a behavior-cloning warm start—but amortizing it across instances, provided the policy generalizes. That proviso is not established here, since the MARL policies were trained on the instance they were evaluated on. A selection layer is attractive precisely because it can exploit this asymmetry; quantifying the benefit requires the equal-time and quality-versus-time analyses specified in L2.

5.5. Limitations and Required Evaluations

The limitations below are stated as specifications rather than acknowledgements, so that each identifies the evaluation that would remove it. The first three are the substantive obstacles to the study's stronger claims.

L1—The routing comparison rests on a single instance and a single run. Reported differences cannot be separated from instance idiosyncrasy or stochastic variation. Removing this requires a corpus of instances varying systematically in size, spatial density, time-window tightness, fleet mix, and disruption level; each solver run with repeated seeds on each; and reporting of mean, dispersion, feasible rate, and wall-clock time, with gaps to optimal or best-known solutions where reference values exist—for which the Solomon time-windowed families [8] and larger capacitated sets [22] are standard reference points—analyzed by rank-based procedures across instances rather than by comparing point values [33].

L2—Search budgets are unequal and not time-matched. Removing this requires equalizing evaluations, repeating the comparison under equal wall-clock time, and reporting quality-versus-time curves so anytime behavior is visible. The MARL side additionally requires reporting demonstration counts for behavior cloning, the curriculum schedule, convergence evidence, hardware, and tuning effort, and separating training from test instances so generalization rather than fit is measured.

L3—The orchestration layer is specified but not evaluated. No selection regret against an oracle, no best-fixed-solver comparison, and no no-fallback ablation are reported, and Equation (10) is not a fitted estimator. Removing this requires running the selector over the corpus of L1 and reporting, per instance, the realized end-to-end objective against three references: an oracle that always picks the best-performing solver, the single best fixed solver across the corpus, and the selector with its feasibility fallback disabled.

L4—Forecasting validation lacks baselines and ablations. No naive or non-recurrent baseline anchors the reported wMAPE, so the intrinsic predictability of the series is unknown; no ablation isolates transfer learning, AutoML selection, the diagnostic rules, or the fallback estimators; errors are aggregate; and each backbone was trained once. Removing this requires a seasonal-naive reference, at least one non-recurrent baseline, per-component ablations, zone-level error distributions, and repeated-seed replication.

L5—Uncertainty propagation is implemented but unmeasured. The upper bound is defined and calibrated (Equation (4)), but point-based and upper-bound-based planning were never compared on fleet size, service level, and cost, and the integrated pipeline was not ablated against a disconnected counterpart. Removing this requires running both planning modes over the corpus of L1.

L6—The corpus is synthetic, calibrated on LaDe rather than on Moroccan operational data. Per Section 3.2, the spatial and temporal structure is generated from the LaDe Hangzhou partition, volumes are generated independently, and the calendar covariates and road network are Moroccan. The forecasting results therefore measure adaptation under a mild spatially induced shift, not transfer between observed cities, and no claim about Casablanca's own demand is supported. Removing this requires operational data from a Moroccan carrier.

L7—Constraint modeling is incomplete. Time windows are soft, with the operating day the only hard temporal bound; all vehicle types share one `driving` matrix, so bicycle infrastructure, pedestrian zones, and access restrictions are unmodeled; and OSRM durations are free-flow. The light-vehicle advantage therefore reflects cost and capacity only. Removing this requires per-profile routing and time-dependent travel times.

L8—Objectives exclude emissions and courier welfare. CO_2 coefficients are carried in the fleet specification but do not enter Equation (8), and workload balance enters only as a mild penalty.

5.6. Future Work

Future work follows the ordering above: the multi-instance, budget-matched, repeated-seed evaluation of L1–L3 is the prerequisite for every other claim, since it is what converts the architecture from a specified mechanism into a measured one. Beyond that, three directions are of particular interest. Constraint handling should be strengthened from step-wise masking toward lookahead-aware or complete-route feasibility enforcement, which the diagnosis in Section 3.7.2 identifies as the proximate cause of the observed compliance failure. Traffic-aware orchestration should replace free-flow durations with time-dependent travel times. Finally, extending the framework to electrified fleets raises coupled routing, charging, and grid-interaction decisions that the present formulation does not represent: recent work shows that intra-day vehicle-to-grid capacity depends jointly on traffic flow and driver behavior [34], and that user-centric guidance over travel and charging decisions can improve both time and energy efficiency [35]. Combining that literature with the CO_2-aware and mixed electric-fleet planning approaches already established in logistics [25,26] is a natural extension of the multi-objective direction identified in L8.

6. Conclusions

This study implemented an orchestration architecture integrating AutoML-based demand forecasting, heterogeneous fleet allocation, and VRPTW routing for last-mile delivery, and used it to compare three metaheuristics and two multi-agent reinforcement learning solvers under one shared, feasibility-aware objective on a common forecast-driven instance.

Four findings are supported. First, the choice among five recurrent forecasting backbones spans only 0.020 of R^2 (best BiLSTM, $R^2 = 0.865$, wMAPE = 29.4%), and bidirectional encoding does not order them consistently, so backbone selection is not the decisive design decision at this stage. Second, feasibility in routing is governed by constraint handling rather than solver paradigm: the two agentic solvers, sharing environment, reward, and budget, produced the most and the least compliant solutions, while the three metaheuristics clustered as fully compliant. Third, distance alone is an invalid efficiency metric under soft windows—MAPPO's shortest route (30.0 km) scores roughly a hundredfold worse than GA's compliant solution (45.4 km) once lateness and shortfall are priced—and the mechanism is identifiable: the mask enforces capacity but not timing, and post-hoc cheapest-insertion repair restores coverage without restoring feasibility. Fourth, the metaheuristic ordering is confounded with search budget, GA having received 5.33× the evaluations of PSO and ACO, and is not evidence of algorithmic superiority.

These conclusions are tightly bounded, and Section 5.5 states the bounds as specifications. The routing comparison rests on one instance solved once per solver, so no statistical claim is made and no general ordering of paradigms is asserted; budgets are unequal and not time-matched; the orchestration layer is specified but not measured against an oracle or a best-fixed solver, leaving the benefit of adaptive selection a hypothesis; forecasting validation lacks baselines, ablations, zone-level distributions, and repeated seeds; the upper bound is calibrated but its planning value is unmeasured; and the corpus is synthetic, generated from the LaDe Hangzhou partition under a documented structure-preserving transformation rather than collected from Moroccan operations.

The contribution we claim is therefore an integrated and fully documented architecture, a mechanistic diagnosis of how forecast-driven routing pipelines can report efficiency gains

that operators would experience as service failures, and a precise specification of the experiments required to establish the performance claims this evidence does not yet support.

Author Contributions: Conceptualization, A.L.; methodology, A.L. and J.B.; software, A.L.; validation, J.B.; formal analysis, A.L.; investigation, A.L.; writing—original draft preparation, A.L.; writing—review and editing, J.B.; supervision, J.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was carried out within the framework of the MILEX project. Mohammed VI Polytechnic University (UM6P) is expected to cover the article processing charge (APC) under the applicable conditions of the MILEX project.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author due to the geolocated nature of the delivery records, whose latitude and longitude coordinates may correspond to sensitive urban locations. The public source from which the corpus is generated is the LaDe dataset, available at https://huggingface.co/datasets/Cainiao-AI/LaDe [32] (accessed on 5 March 2026); the generation procedure and its parameters (Section 3.2, Table 2) and the benchmark routing instance (Table 4) are documented in full, so that an equivalent corpus and instance can be regenerated independently.

Acknowledgments: This work was conducted as part of the MILEX project and greatly benefited from the support of Hassan II University of Casablanca, Ecole Centrale Casablanca, and Mohammed VI Polytechnic University. The collaborative and constructive involvement of these institutions played an important role in the successful realization of this work. The authors thank the anonymous reviewers, whose questions on data provenance, computational budgets, and the feasibility of the reported agentic solutions led directly to the corrections and disclosures in Sections 3.2, 3.7.2 and 3.9.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ACO	Ant Colony Optimization
AOI	Area of Interest
AutoML	Automated Machine Learning
BiGRU	Bidirectional Gated Recurrent Unit
BiLSTM	Bidirectional Long Short-Term Memory
GA	Genetic Algorithm
GRU	Gated Recurrent Unit
HPO	Hyperparameter Optimization
LSTM	Long Short-Term Memory
MAD	Moroccan Dirham
MADQN	Multi-Agent Deep Q-Network
MAE	Mean Absolute Error
MAPPO	Multi-Agent Proximal Policy Optimization
MARL	Multi-Agent Reinforcement Learning
OSM	OpenStreetMap
OSRM	Open Source Routing Machine
PSO	Particle Swarm Optimization
RL	Reinforcement Learning
RMSE	Root Mean Square Error
SLA	Service Level Agreement
VDN	Value Decomposition Network

VRPTW	Vehicle Routing Problem with Time Windows
wMAPE	Weighted Mean Absolute Percentage Error

References

1. Boysen, N.; Fedtke, S.; Schwerdfeger, S. Last-mile delivery concepts: A survey from an operational research perspective. *OR Spectr.* **2021**, *43*, 1–58. [CrossRef]
2. Silva, V.; Amaral, A.; Fontes, T. Sustainable urban last-mile logistics: A systematic literature review. *Sustainability* **2023**, *15*, 2285. [CrossRef]
3. Richey, R.G.; Chowdhury, S.; Davis-Sramek, B.; Giannakis, M.; Dwivedi, Y.K. Artificial intelligence in logistics and supply chain management: A primer and roadmap for research. *J. Bus. Logist.* **2023**, *44*, 532–549. [CrossRef]
4. Mamede, F.P.; da Silva, R.F.; de Brito Junior, I.; Yoshizaki, H.T.Y.; Hino, C.M.; Cugnasca, C.E. Deep learning and statistical models for forecasting transportation demand: A case study of multiple distribution centers. *Logistics* **2023**, *7*, 86. [CrossRef]
5. Xia, K.; Lin, L.; Zhang, X.; Wang, H.; Wang, S.; He, T. A transferable spatio-temporal learning framework for cross-city logistics demand prediction. In Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '25), Toronto, ON, Canada, 3–7 August 2025; pp. 5071–5082. [CrossRef]
6. Alsharef, A.; Aggarwal, K.; Sonia; Kumar, M.; Mishra, A. Review of ML and AutoML solutions to forecast time-series data. *Arch. Comput. Methods Eng.* **2022**, *29*, 5297–5311. [CrossRef] [PubMed]
7. Schroeder, M.; Lodemann, S. A systematic investigation of the integration of machine learning into supply chain risk management. *Logistics* **2021**, *5*, 62. [CrossRef]
8. Solomon, M.M. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* **1987**, *35*, 254–265. [CrossRef]
9. Chen, G.; Gao, J.; Chen, D. Research on vehicle routing problem with time windows based on improved genetic algorithm and ant colony algorithm. *Electronics* **2025**, *14*, 647. [CrossRef]
10. Lesch, V.; Koenig, M.; Kounev, S.; Stein, A.; Krupitzer, C. Tackling the rich vehicle routing problem with nature-inspired algorithms. *Appl. Intell.* **2022**, *52*, 9476–9500. [CrossRef]
11. Gao, Y.; Wu, H.; Wang, W. A hybrid ant colony optimization with fireworks algorithm to solve capacitated vehicle routing problem. *Appl. Intell.* **2023**, *53*, 7326–7342. [CrossRef]
12. Ky Phuc, P.N.; Phuong Thao, N.L. Ant colony optimization for multiple pickup and multiple delivery vehicle routing problem with time window and heterogeneous fleets. *Logistics* **2021**, *5*, 28. [CrossRef]
13. Yadav, P.; Mishra, A.; Kim, S. A comprehensive survey on multi-agent reinforcement learning for connected and automated vehicles. *Sensors* **2023**, *23*, 4710. [CrossRef] [PubMed]
14. Yu, C.; Velu, A.; Vinitsky, E.; Gao, J.; Wang, Y.; Bayen, A.; Wu, Y. The surprising effectiveness of PPO in cooperative multi-agent games. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24611–24624. [CrossRef]
15. Ming, Y.-J.; Shang, Q.-X.; Qian, B.; Hu, R. Multi-agent deep Q-learning algorithm integrated with ant colony optimization for solving dynamic feeder vehicle routing problem. In *Advanced Intelligent Computing Technology and Applications (ICIC 2025)*; Lecture Notes in Computer Science; Huang, D.S., Pan, Y., Chen, W., Li, B., Eds.; Springer: Singapore, 2025; Volume 15854, pp. 1–12. [CrossRef]
16. Pan, W.; Liu, S. Deep reinforcement learning for the dynamic and uncertain vehicle routing problem. *Appl. Intell.* **2023**, *53*, 405–422. [CrossRef]
17. Bogyrbayeva, A.; Meraliyev, M.; Mustakhov, T.; Dauletbayev, B. Machine learning to solve vehicle routing problems: A survey. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 4754–4772. [CrossRef]
18. Zhou, L.; Li, C.; Hu, C.; Du, J. Parcel locker location problem with selectable volume sizes and heterogeneous customers in the last mile delivery. *Transp. Lett.* **2023**, *16*, 1140–1154. [CrossRef]
19. Ostermeier, M.; Heimfarth, A.; Hübner, A. Cost-optimal truck-and-robot routing for last-mile delivery. *Networks* **2021**, *79*, 364–389. [CrossRef]
20. Sina Mohri, S.; Ghaderi, H.; Nassir, N.; Thompson, R.G. Crowdshipping for sustainable urban logistics: A systematic review of the literature. *Transp. Res. Part E Logist. Transp. Rev.* **2023**, *178*, 103289. [CrossRef]
21. Orozco Torres, J.A.; Medina Santiago, A.; García-Martínez, J.R.; López-Zapata, B.Y.; Mijangos López, J.A.; Rincón Zapata, O.J.; Avitia López, J.A. A data-driven approach using recurrent neural networks for material demand forecasting in manufacturing. *Logistics* **2025**, *9*, 130. [CrossRef]
22. Uchoa, E.; Pecin, D.; Pessoa, A.; Poggi, M.; Vidal, T.; Subramanian, A. New benchmark instances for the capacitated vehicle routing problem. *Eur. J. Oper. Res.* **2017**, *257*, 845–858. [CrossRef]
23. Marinakis, Y.; Marinaki, M.; Migdalas, A. Particle swarm optimization for the vehicle routing problem: A survey and a comparative analysis. In *Handbook of Heuristics*; Martí, R., Pardalos, P.M., Resende, M.G.C., Eds.; Springer: Cham, Switzerland, 2018; pp. 1163–1196. [CrossRef]

24. Nadime, K.L.; Benhra, J.; Benabbou, R.; Mouatassim, S. A blockchain-powered framework for traceable and secure pharmaceutical delivery with crowdsourced logistics. In *Innovative Intelligent Industrial Production and Logistics. IN4PL 2024*; Communications in Computer and Information Science; Dassisti, M., Madani, K., Panetto, H., Eds.; Springer: Cham, Switzerland, 2025; Volume 2372, pp. 367–384. [CrossRef]
25. Ejjanfi, M.-A.; Benhra, J. Flexible CO_2-aware routing with real-time smart locker adjustments. In *Innovative Intelligent Industrial Production and Logistics*; Springer: Cham, Switzerland, 2026; pp. 242–262. [CrossRef]
26. Al-dal'ain, R.; Celebi, D. Planning a mixed fleet of electric and conventional vehicles for urban freight with routing and replacement considerations. *Sustain. Cities Soc.* **2021**, *73*, 103105. [CrossRef]
27. Guerreiro Augusto, M.; Maas, J.; Kosch, M.; Henke, M.; Küster, T.; Straube, F.; Albayrak, S. Autonomous van and robot last-mile logistics platform: A reference architecture and proof of concept implementation. *Logistics* **2025**, *9*, 10. [CrossRef]
28. Abideen, A.Z.; Sundram, V.P.K.; Pyeman, J.; Othman, A.K.; Sorooshian, S. Digital twin integrated reinforced learning in supply chain and logistics. *Logistics* **2021**, *5*, 84. [CrossRef]
29. Oucheikh, R.; Löfström, T.; Ahlberg, E.; Carlsson, L. Rolling cargo management using a deep reinforcement learning approach. *Logistics* **2021**, *5*, 10. [CrossRef]
30. Arishi, A.; Krishnan, K. A multi-agent deep reinforcement learning approach for solving the multi-depot vehicle routing problem. *J. Manag. Anal.* **2023**, *10*, 493–515. [CrossRef]
31. Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; Meger, D. Deep reinforcement learning that matters. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32, pp. 3207–3214. [CrossRef]
32. Wu, L.; Wen, H.; Hu, H.; Mao, X.; Xia, Y.; Shan, E.; Zhen, J.; Lou, J.; Liang, Y.; Yang, L.; et al. LaDe: The first comprehensive last-mile delivery dataset from industry. In Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), Barcelona, Spain, 25–29 August 2024; pp. 5991–6002. Available online: https://huggingface.co/datasets/Cainiao-AI/LaDe (accessed on 5 March 2026).
33. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
34. Ke, S.; Zhang, K.; Mai, W.; Guo, R.; He, S.; Tian, J.; Chung, C.Y. Maximizing intraday V2G feasible capacity of EVs: A cross-disciplinary approach with traffic flow and prospect theory. *IEEE Trans. Transp. Electrif.* **2026**, *12*, 5078–5091. [CrossRef]
35. Fan, P.; Bu, S.; Wen, Y.; Ke, S.; Li, S.; Fang, S. Electric vehicle user-centric guidance on travel and charging decisions to enhance time and energy efficiency. *IEEE Trans. Transp. Electrif.* **2026**, *in press*. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.