INSPIRE SMART SYSTEMS

Inspire Smart Systems Journal
Vol. 1, No. 1, 2026
https://doi.org/10.65718/inspireSS.2026.3002

InspireQuill.org
Inspire Academic Press Inc. (Canada)

Research Paper

Reinforcement learning approach based on Proximal Policy Optimization algorithm for efficient last mile delivery using Smart Lockers

Mohamed-Ali Ejjanfi [1] and Jamal Benhra [1, *]

[1] LARILE, ENSEM, University Hassan II of Casablanca, Morocco
[*] Corresponding Author: jbenhra@ensem.ac.ma

Abstract

The Vehicle Routing Problem (VRP) is a core logistics challenge where routing choices drive cost and service quality. Reinforcement Learning (RL), and in particular Proximal Policy Optimization (PPO), now rivals traditional metaheuristics by learning adaptive routing policies that transfer across instances. We compare PPO with the well-established Ant Colony Optimization (ACO) on city-based benchmarks. Extensive experiments on synthetic and real datasets substantiate these findings. Results show PPO delivers comparable or superior routing efficiency and, after training, orders-of-magnitude faster inference, whereas ACO stays competitive on static VRPs but falters in dynamic, large-scale settings. Our analysis underscores trade-offs between learning-based and search-based methods, highlighting scalability, computation time, and adaptability, and offers guidance for future intelligent logistics optimization.

Keywords Last-Mile Delivery, Reinforcement Learning (RL), Proximal Policy Optimization (PPO), Metaheuristics, Ant Colony Optimization (ACO), Smart locker, Logistics Optimization.

1. Introduction

The Vehicle Routing Problem (VRP) underpins modern e-commerce and urban logistics [1]. It asks how a fleet can serve dispersed customers and return to base while minimizing distance, time, or emissions [2]. Because the VRP is NP-hard, exact solvers scale poorly, so decades of research have favored heuristics and metaheuristics that trade a little optimality for a lot of tractability [3].
Two recent trends now stretch those classical methods: unattended smart-locker delivery, which decouples customer presence from drop-offtime [4], and high-frequency dynamism driven by real-time traffic, late orders, and crowd-sourced fleets [5]. These factors enlarge instances, add constraints (locker capacity, pickup windows, energy budgets), and demand rapid re-optimization [6].

Academic Editor:
Nazeeruddin Mohammad

Received: 15/05/2025
Revised: 22/07/2025
Accepted: 14/08/2025
Published: 01/01/2026

Citation
Ejjanfi, M., Benhra, J. (2026). Reinforcement learning approach based on Proximal Policy Optimization algorithm for efficient last mile delivery using Smart Lockers. *Inspire Smart Systems Journal, 1*(1), 9-25.

Ant Colony Optimization (ACO) remains effective for static VRPs thanks to its exploration, exploitation balance [19], but it must restart whenever inputs shift, limiting real-time use [20]. Reinforcement learning, particularly Proximal Policy Optimization (PPO), offers an alternative: it learns an offline policy and then generates routes in milliseconds, amortizing training cost over many instances [21-22].

Smart-locker networks compound the challenge. While they cut failed deliveries and idling [23], their capacity limits and pickup-time preferences reshape route feasibility [24]. Encoding such nuances in an RL state space lets a PPO agent internalize locker availability, whereas pheromone-based heuristics need bespoke tuning or repair operators [25-26].

Our study therefore: (1) frames last-mile VRP with smart lockers for Casablanca, Marrakech, and Tangier as a sequential decision process [27]; (2) develops city-specific PPO agents and a tuned ACO baseline under identical metrics and fleets [28]; and (3) benchmarks them, showing PPO cuts distance by up to 17 % with sub-second inference, while ACO remains faster when no pretrained model exists [29].

By situating these findings within the drive for greener, faster, and more autonomous logistics, we underscore how learning-based solvers can complement, rather than replace, established metaheuristics [30-31], and how Industry 4.0 enablers such as blockchain-secured traceability and AI-driven analytics are already reshaping last-mile operations [32].

2. State of art

2.1. Ant Colony Optimization (ACO)

2.1.1. Principle

Artificial "ants" build a route step-by-step, choosing the next customer by combining the pheromone strength on an edge, an accumulated memory of past good decisions, and a heuristic such as inverse distance [13]. Pheromone deposited on high-quality tours is reinforced, while global evaporation steadily lowers all trails to avoid early stagnation [13]. This positive-feedback search, "follow stronger trails, reinforce the best paths" [14], has proved effective across many VRP variants and constraint sets [16].

2.1.2. Formulations

Pheromone update [13]

$$\tau_{ij}(t+1) = (1-p)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}\,(t)$$

- $\tau_{ij}(t)$ is the pheromone level on edge (i,j) at iteration t.
- p ∈(0,1) is the pheromone evaporation rate; larger ρ implies faster evaporation.
- $\Delta\tau_{ij}^{k}$ is the amount of pheromone deposited by ant k on (i, j) during iteration t.
- m is the number of ants

Equation (1) balances reinforcement with "forgetting," so only edges in consistently good solutions accumulate pheromone.

Probabilistic transition rule[13]

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha}\,[\eta_{ij}]^{\beta}}{\sum_{l\,\in N_i^k}[\tau_{il}(t)]^{\alpha}\,[\eta_{il}]^{\beta}}$$

- $p_{ij}^{k}(t)$ is the probability that ant k at node iii moves next to node jjj at iteration t.
- $\tau_{ij}(t)$ is the current pheromone level for edge (i, j).
- η_{ij} is a heuristic factor (often $\eta_{ij} = \frac{1}{d_{ij}}$, the inverse of the distance).
- $\alpha \geq 0$ is the exponent controlling how strongly ants follow pheromones.
- $\beta \geq 0$ is the exponent controlling how strongly ants follow the heuristic.
- η_{il} is the set of feasible nodes for ant k from node i.

If α is large, ants heavily prioritize exploiting learned pheromone trails.
If β is large, ants prioritize exploring edges with higher heuristic value.

2.1.3. Strengths & limits

ACO is flexible, easy to parallelize and often delivers near-optimal routes without problem-specific learning, yet performance hinges on careful tuning and re-running when input data change [13].

2.2. Proximal Policy Optimization (PPO)

2.2.1. Principle

PPO is a policy-gradient RL algorithm that improves a stochastic policy while clipping each update to stay close to the previous policy, preventing destabilizing jumps [10]. A VRP episode is modelled as an MDP where the state encodes depot/vehicle position and unserved customers, the action selects the next customer (or depot return), and the reward is the negative travel cost so maximizing cumulative reward minimizes distance [12].

2.2.2. PPO Policy Equation

Clipped objective [10]:
PPO typically represents its policy as (a ∣ s), the probability of taking action a given state s under parameters θ. The update rule clips the probability ratio between old and new policies, ensuring changes are not too drastic [10]. The clipped objective can be written as:

$$L^{CLIP}(\theta) = E_t\left[\min\left(r_t(\theta)\hat{A}, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}\right]\right.$$

- $r_t(\theta) = \frac{\pi_\theta(st)}{\pi_{\theta_{old}}(st)}$ is the probability ratio between new and old policies.
- Â is an estimator of the advantage (how much better an action is compared to average).
- ϵ is a small hyperparameter (e.g., 0.1 or 0.2) controlling the clipping range.

The function min (·,·) forces the optimizer to pick the clipped or unclipped objective, whichever is lower, thereby discouraging updates that move rt(θ) outside [1−ϵ, 1+ϵ]. This reduces variance in training and helps maintain stable performance. In VRP, states correspond to the vehicle's current position and the unvisited customers; actions choose the next customer to serve; and rewards are usually negative distance or cost. By sampling many simulated VRP episodes, PPO refines θ to yield better routes over time.

2.2.3. Advantages for VRP

The approach is data-driven, capable of discovering routing patterns that go beyond hand-crafted heuristics [10]. It is scalable, effectively handling hundreds of customers by leveraging neural-network policies [9]. With fast inference, the model can generate high-quality routes in milliseconds once trained, making it well-suited for dynamic dispatch scenarios [12]. Additionally, the policy is adaptable, allowing for periodic fine-tuning as new traffic or order data become available [12].

ACO offers interpretable, pheromone-guided search with strong anytime performance on static instances [13, 4]. PPO trades a higher offline training cost for real-time inference and on-line adaptability [12, 9]. Choosing or hybridizing these methods depends on the frequency of instance changes, latency constraints, and available compute.

3. Benchmark Methodology for PPO vs ACO Comparison

To concretely compare PPO and ACO on VRP, we consider a benchmarking study across three city-based instances: Casablanca, Marrakech, and Tangier. These instances represent realistic logistics scenarios in different cities, allowing us to evaluate how each algorithm performs in distinct settings. Below we outline the methodology used for this comparative experiment:

Problem Setup: Each city instance consists of a set of delivery locations (customers) spread around the city and a central depot. For simplicity, we assume a capacitated VRP: a fixed fleet of vehicles (with uniform capacity) starts at the depot and must serve all customer demands, returning to the depot when routes are completed. The goal is to minimize the total distance traveled by all vehicles while serving all customers. Distances between locations are based on real coordinates and road distances for each city (ensuring a realistic, non-uniform distance matrix) and explicitly factor in relevant geographic and cadastral considerations. Accordingly, Tangier's Road network incorporates notable elevation changes (affecting travel times and fuel consumption), Marrakech features predominantly flat terrain (leading to more consistent travel speeds), and Casablanca, a larger metropolis, entails more complex routing due to its dense infrastructure and frequent congestion. As a result, Casablanca has a slightly higher number of customers in its instance compared to Marrakech and Tangier in our setup, reflecting its greater urban sprawl and heightened logistical demands.

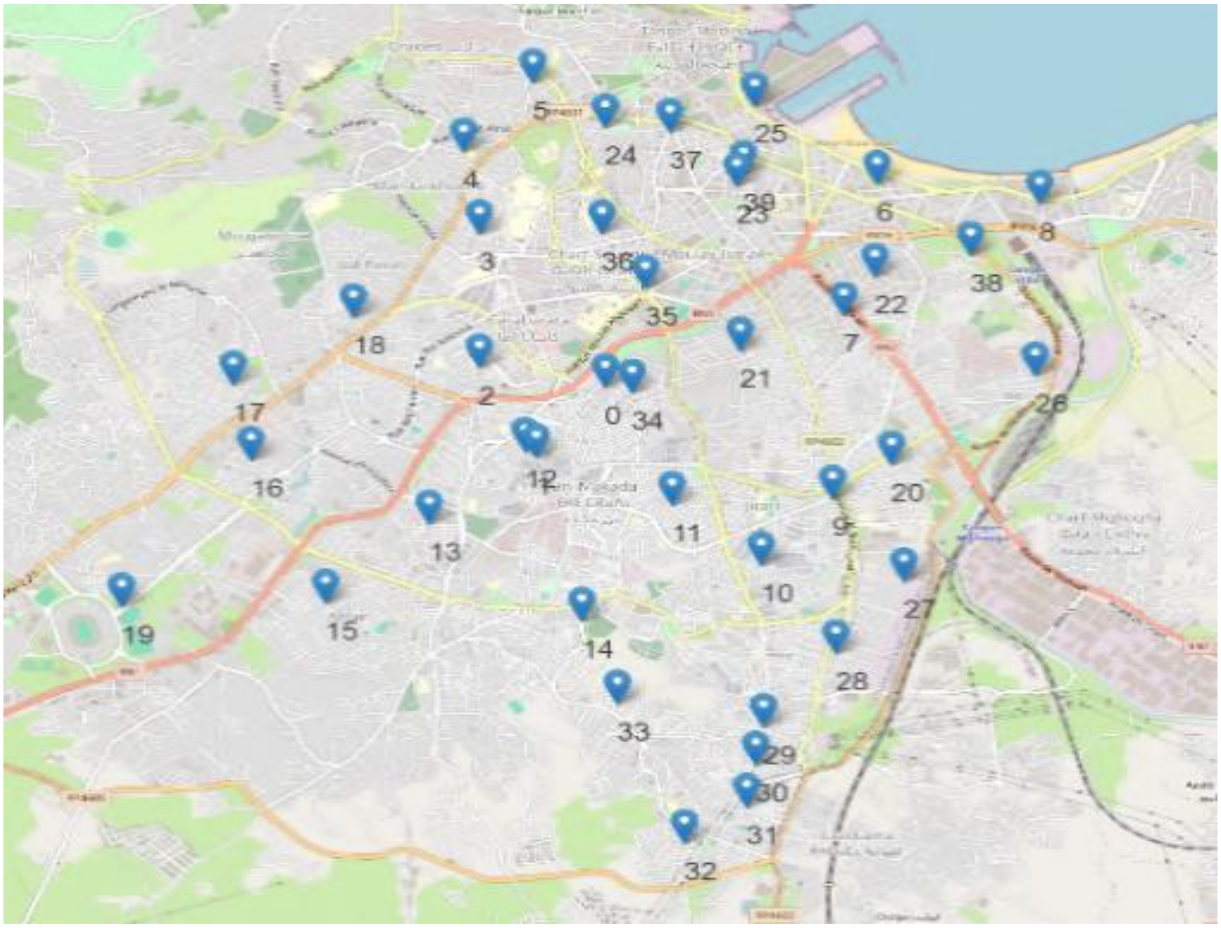


Figure 1. Spatial Distribution of Parcel Lockers: A Geographic Representation in Tanger.

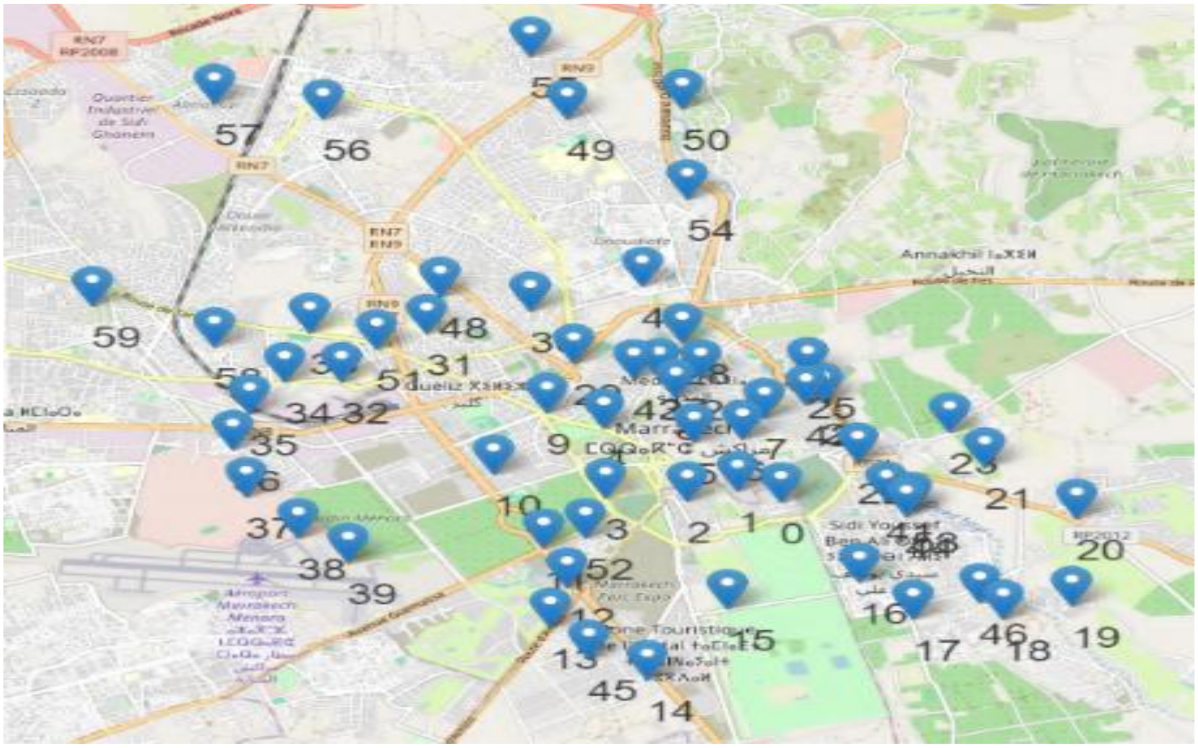


Figure 2. Spatial Distribution of Parcel Lockers: A Geographic Representation in Marrakech.

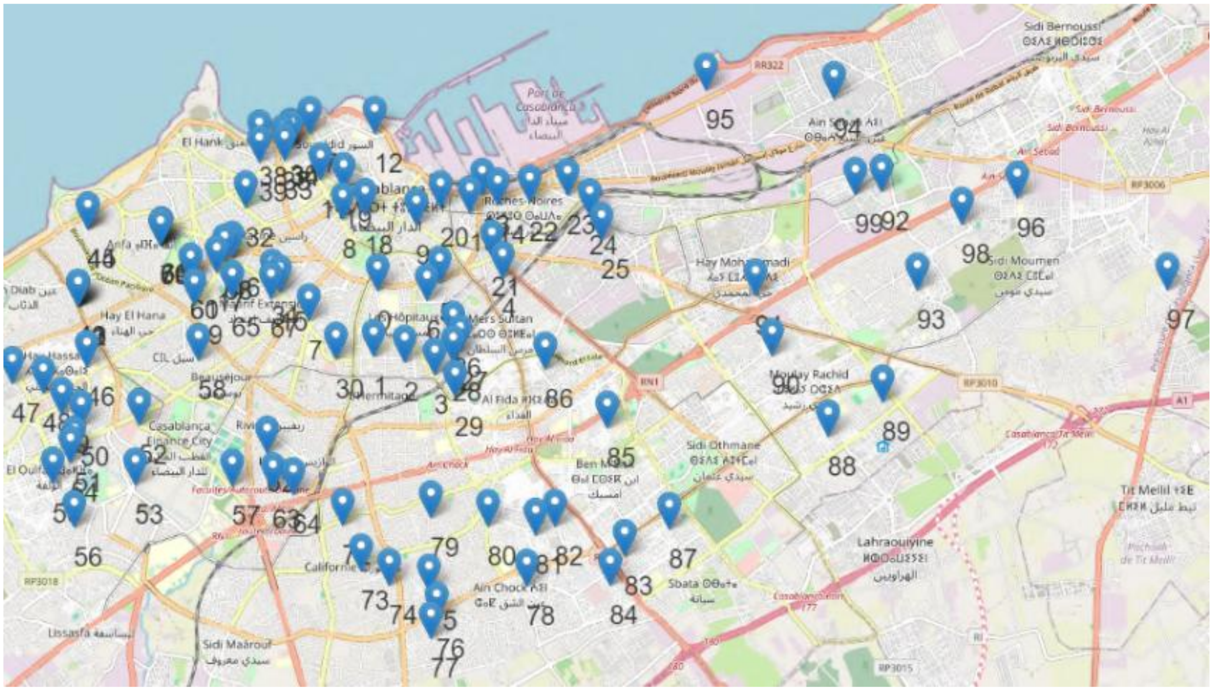


Figure 3. Spatial Distribution of Parcel Lockers: A Geographic Representation in Casablanca.

3.1. Ant Colony Optimization (ACO) Implementation

We adopted a conventional ACO approach for the Vehicle Routing Problem (VRP). In our implementation, each "ant" incrementally builds a solution by selecting the next customer to visit according to two main factors:

- Pheromone Trails: A measure indicating how favorable an edge (between two locations) has been in previous solutions, raised to the power α.
- Heuristic Value: The inverse of distance between the current location and a potential next customer, raised to the power β.
- Once an ant's vehicle runs out of capacity or it finishes serving all assigned customers, it returns to the depot and may switch to a new vehicle if available. After a set of ants (i.e., a generation) have constructed their solutions, pheromone values are updated:
- Evaporation: Pheromones across all edges are scaled by $(1-\rho)$.
- Deposition: Each ant deposits additional pheromone on the edges it traversed, with amounts inversely proportional to the ant's total route cost. Hence, routes with lower cost reinforce higher pheromone levels, guiding subsequent solutions.

For larger city instances (e.g., Casablanca with 100 customers), we typically run around 60 ants for 200 iterations. These parameters are consistent with existing ACO literature [7] and yield a reasonable balance between runtime and solution quality.

3.2. Proximal Policy Optimization (PPO) Implementation

Our solver learns to reassign customers to vehicle clusters; a cost-aware 3-Opt routine then turns every cluster into a route. The objective is total travelled distance.

3.2.1. Training Phase

State information consists of the distance matrix, a one-hot assignment matrix (customer → vehicle), vehicle capacities and customer demands.

- The policy is split into two convolutional actors:
    1. Node-selection actor. Two 3 × 3 Conv2D layers (27 kernels each) followed by ReLU and a fully connected layer output a categorical distribution over N customers.
    2. Cluster-assignment actor. The same 2-layer CNN backbone feeds a fully connected layer that produces logits for K vehicle clusters; a capacity mask zero out clusters that would overflow.

The critic shares this two-layer CNN backbone but ends with a single linear unit that estimates the state value. All networks process a 3-D "image" made of the distance matrix (1 channel) stacked with K binary cluster maps, letting the CNN exploit spatial regularities in the adjacency structure.

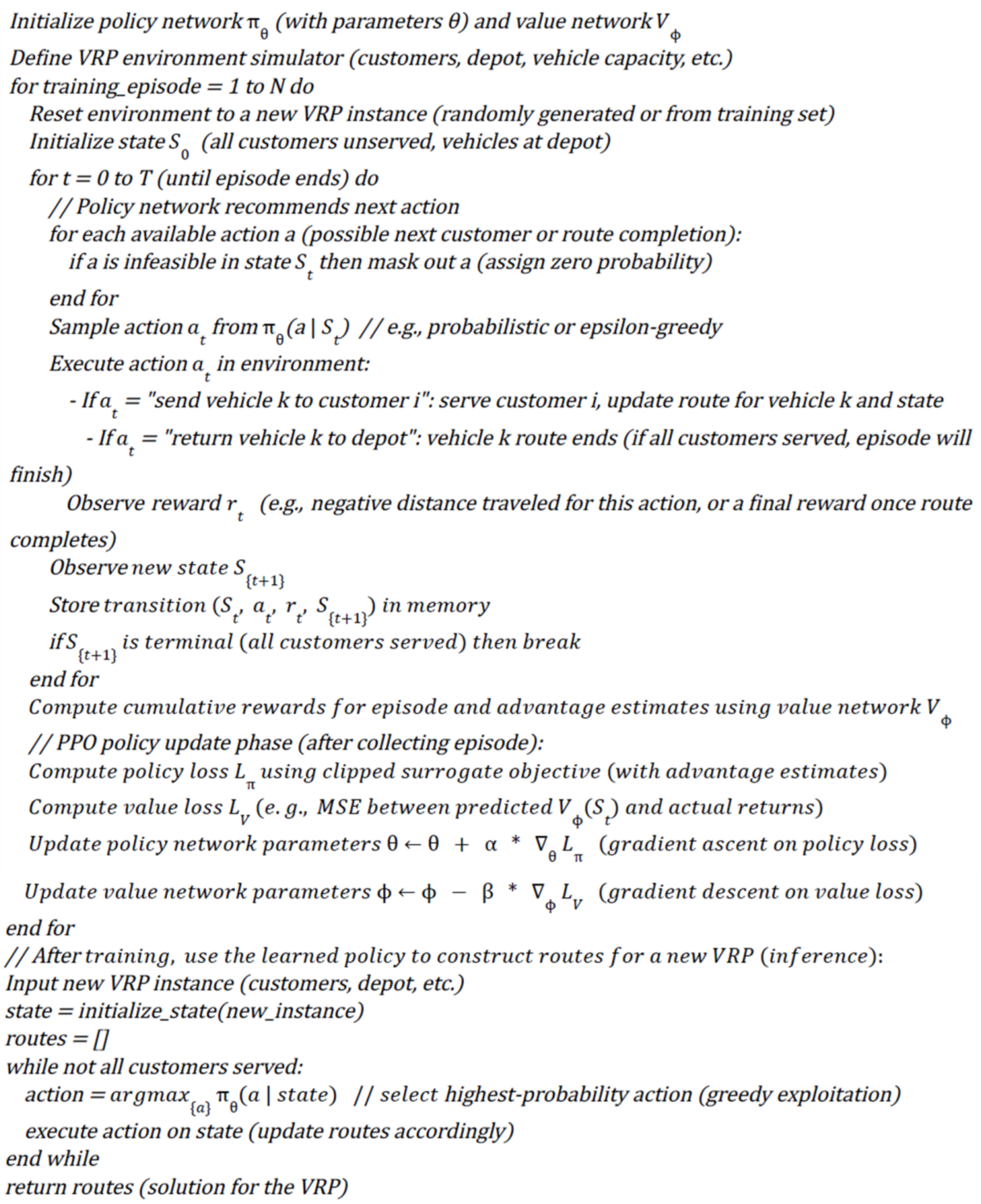
After each reassignment the environment reruns 3-Opt on every cluster and gives the agent a reward equal to the distance reduction divided by a constant scale. Parameters are updated with the clipped PPO objective and Generalized Advantage Estimation (GAE) [11].

3.2.2. Solution Phase

When training stops the best assignment seen is refined by a short random reassignment search plus one more 3-Opt round, then converted into final vehicle routes. This inference step still calls 3-Opt but with a small, fixed budget, so it is far faster than running a full metaheuristic from scratch.

To clarify how the PPO approach works for VRP, we provide a simplified pseudo-code for the training and solution process. This outlines how the reinforcement learning agent learns to construct routes and then uses that learned policy.

Pseudo-code for PPO-based VRP solver



```
Initialize policy network $\pi_\theta$ (with parameters $\theta$) and value network $V_\phi$
Define VRP environment simulator (customers, depot, vehicle capacity, etc.)
for training_episode = 1 to N do
  Reset environment to a new VRP instance (randomly generated or from training set)
  Initialize state $S_0$ (all customers unserved, vehicles at depot)
  for t = 0 to T (until episode ends) do
    // Policy network recommends next action
    for each available action a (possible next customer or route completion):
      if a is infeasible in state $S_t$ then mask out a (assign zero probability)
    end for
    Sample action $a_t$ from $\pi_\theta(a \mid S_t)$ // e.g., probabilistic or epsilon-greedy
    Execute action $a_t$ in environment:
      - If $a_t$ = "send vehicle k to customer i": serve customer i, update route for vehicle k and state
        - If $a_t$ = "return vehicle k to depot": vehicle k route ends (if all customers served, episode will
finish)
      Observe reward $r_t$ (e.g., negative distance traveled for this action, or a final reward once route
completes)
    Observe new state $S_{\{t+1\}}$
    Store transition ($S_t$, $a_t$, $r_t$, $S_{\{t+1\}}$) in memory
    if $S_{\{t+1\}}$ is terminal (all customers served) then break
  end for
  Compute cumulative rewards for episode and advantage estimates using value network $V_\phi$
  // PPO policy update phase (after collecting episode):
  Compute policy loss $L_\pi$ using clipped surrogate objective (with advantage estimates)
  Compute value loss $L_V$ (e. g., MSE between predicted $V_\phi(S_t)$ and actual returns)
  Update policy network parameters $\theta \leftarrow \theta + \alpha * \nabla_\theta L_\pi$ (gradient ascent on policy loss)
  Update value network parameters $\phi \leftarrow \phi - \beta * \nabla_\phi L_V$ (gradient descent on value loss)
end for
// After training, use the learned policy to construct routes for a new VRP (inference):
Input new VRP instance (customers, depot, etc.)
state = initialize_state(new_instance)
routes = []
while not all customers served:
  action = $argmax_{\{a\}} \pi_\theta(a \mid state)$ // select highest-probability action (greedy exploitation)
  execute action on state (update routes accordingly)
end while
return routes (solution for the VRP)
```

4. Benchmark Results and Discussion

We tested two algorithms, Ant Colony Optimization (ACO) and Proximal Policy Optimization (PPO), on three city-based VRP instances: Casablanca (largest), Marrakech (medium), and Tangier (smallest), with node index 0 serving as the depot in every case.

Table 1. Morocco CVRP Cases.

Instance	City	Vehicles	Customers	Capacity vector	Demand vector
N40k5	Tanger	5	39	[43, 47, 39, 56, 47]	[5, 6, 7, 5, 9, 3, 6, 8, 6, 9, 3, 4, 1, 8, 5, 9, 5, 7, 4, 8, 6, 10, 5, 2, 8, 7, 3, 2, 7, 9, 10, 3, 2, 10, 5, 4, 6, 5, 4]
N60k7	Marrakech	6	59	[48, 57, 39, 59, 37, 49, 40]	[7, 10, 7, 1, 7, 3, 6, 3, 2, 9, 3, 6, 5, 7, 7, 1, 6, 2, 9, 3, 7, 3, 10, 2, 6, 2, 1, 6, 2, 10, 8, 3, 1, 2, 9, 3, 10, 5, 5, 10, 10, 4, 9, 9, 4, 8, 10, 10, 1, 10, 2, 3, 5, 5, 1, 6, 5, 3, 9]
N100k9	Casablanca	9	99	[85, 69, 55, 50, 84, 75, 58, 65, 60]	[4, 8, 10, 8, 5, 10, 7, 10, 7, 6, 8, 3, 10, 1, 2, 5, 10, 3, 8, 3, 8, 2, 3, 7, 9, 2, 7, 2, 5, 10, 7, 3, 9, 10, 10, 2, 4, 10, 5, 3, 7, 5, 10, 10, 6, 6, 6, 6, 8, 6, 2, 3, 9, 6, 1, 4, 9, 1, 5, 8, 2, 4, 3, 9, 7, 1, 9, 2, 4, 3, 2, 4, 6, 10, 10, 1, 5, 8, 7, 9, 3, 8, 4, 8, 4, 2, 3, 3, 7, 7, 5, 10, 3, 10, 8, 9, 8, 4, 10]

For PPO, hyperparameter tuning was conducted using a grid search strategy implemented via Itertools, allowing us to explore a predefined range of values for each parameter in a systematic and exhaustive manner. This approach enabled a direct comparison of multiple configurations while keeping the search space interpretable and manageable. The best-performing configuration was:

- num_clusters = len(capacities),
- actor_lr = 1e-5,
- critic_lr = 1e-5,
- gamma = 0.99,
- lam = 0.95,
- clip_epsilon = 0.2,
- kl_target = 0.01,
- beta = 1.0,
- entropy_coeff= 0.1,
- num_kernels = 27,
- max_steps = 1000,
- reward_scale = 10.0,
- seed = 42.

For ACO, we similarly used grid search via itertools to tune key parameters by testing all combinations from a fixed candidate set. This allowed us to identify the parameter configuration that produced the best solution quality across instances. The selected parameters were:

- num_of_generations = 200,
- ant = number of smart lockers,
- alpha = 0.2,
- beta = 10,
- rho = 0.9.

Here we summarize them and suggest potential optimizations to mitigate weaknesses:

Table 2. Result Summary.

Instance	**City**	**ACO Distance**	**ACO Time (sec)**	**PPO Distance**	**PPO Time (sec)**
n100k9	Casablanca	234.228	1886.727	194.176	3520.23
N60k7	Marrakech	115.778	812.572	114.35	1256.72
N40k5	Tanger	69.293	440.267	60.025	870.173

Benchmarking on three Moroccan VRP instances shows a clear distance advantage for PPO over ACO:

- Tangier (39 customers) 69.29 km → 60.03 km
- Marrakech (59 customers) 115.78 km → 114.35 km
- Casablanca (99 customers) 234.23 km → 194.18 km

The largest reduction (-17 %) appears in Casablanca, where the learned policy exploits repeated spatial patterns; the smallest gap (-1 %) occurs in Marrakech, suggesting that ACO's pheromone search already captures most of the structure there. Computationally, ACO completes in 10–30 min, while PPO needs roughly twice as long because training is executed per instance. When a previously trained PPO is reused for inference only, generation time falls to milliseconds, reversing the timing gap [18]. Both methods are sensitive to hyper-parameters. ACO's performance shifts with α,β,ρ; PPO reacts strongly to the learning rate and clip coefficient. Grid search delivered robust settings, α = 0.2, β = 10, ρ = 0.9 for ACO and lr = 10^{-5}, clip = 0.2 for PPO, but small changes can alter the distance–time trade-off.

Finally, ACO remains fully instance-agnostic, whereas PPO's quality depends on training data similarity; a significant geography or demand shift would require retraining or fine-tuning.

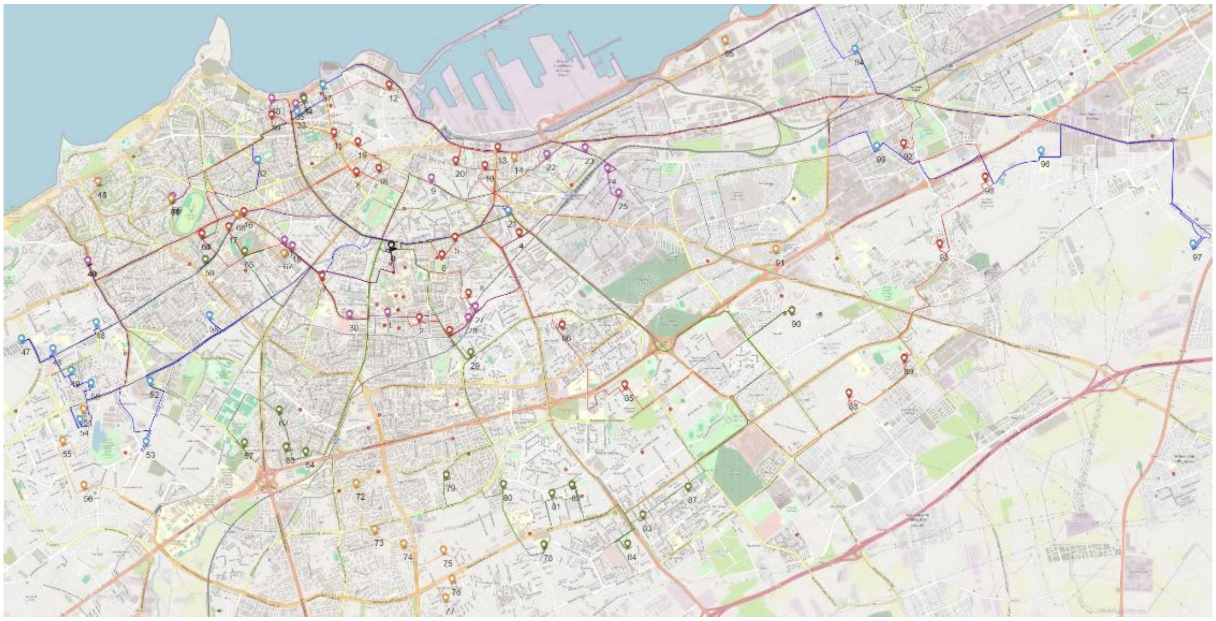


Figure 4. PPO-Optimized Route for VRP in Casablanca

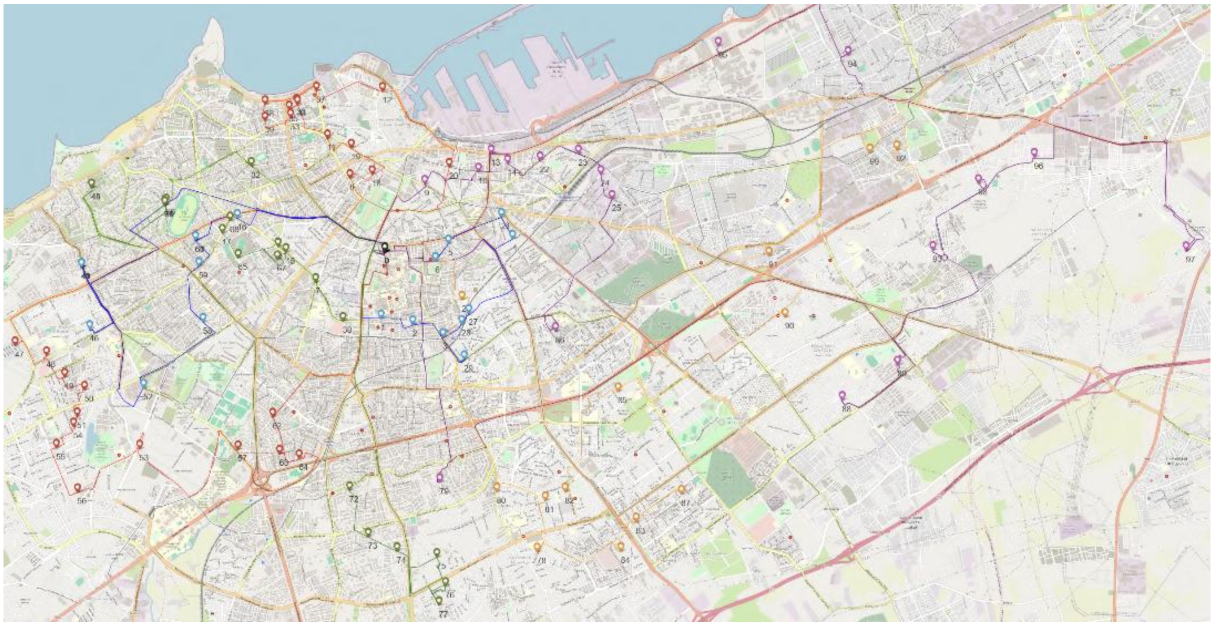


Figure 5. ACO-Optimized Route for VRP in Casablanca

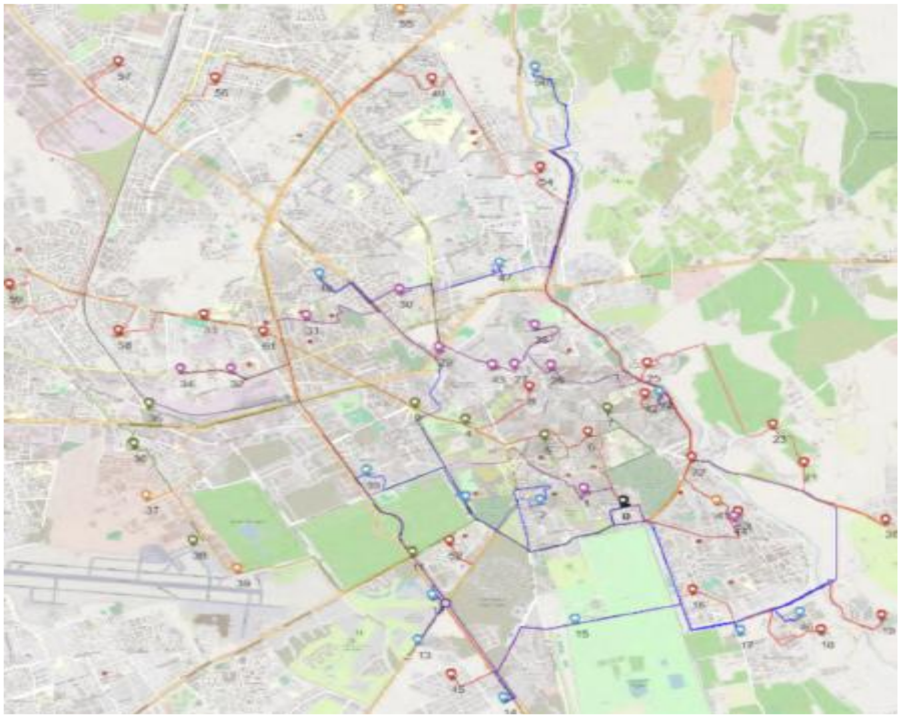


Figure 6. Comparison between PPO and ACO optimization: (**a**) PPO-Optimized Route for VRP in Marrakech

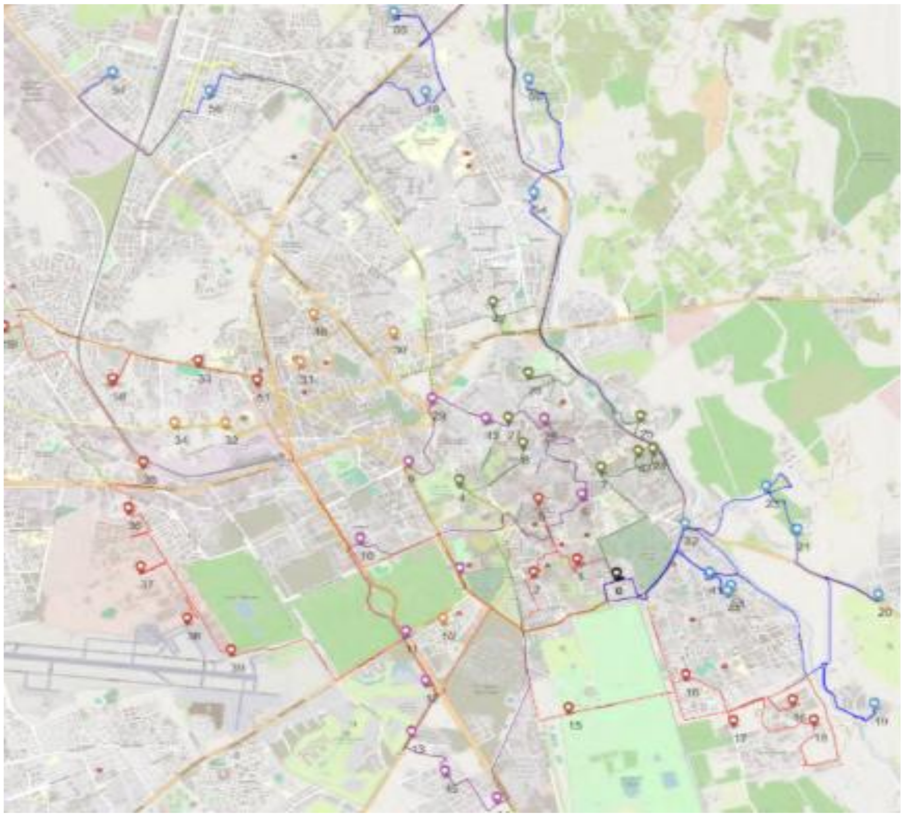


Figure 6. Comparison between PPO and ACO optimization: (**b**) ACO-Optimized Route for VRP in Marrakech.

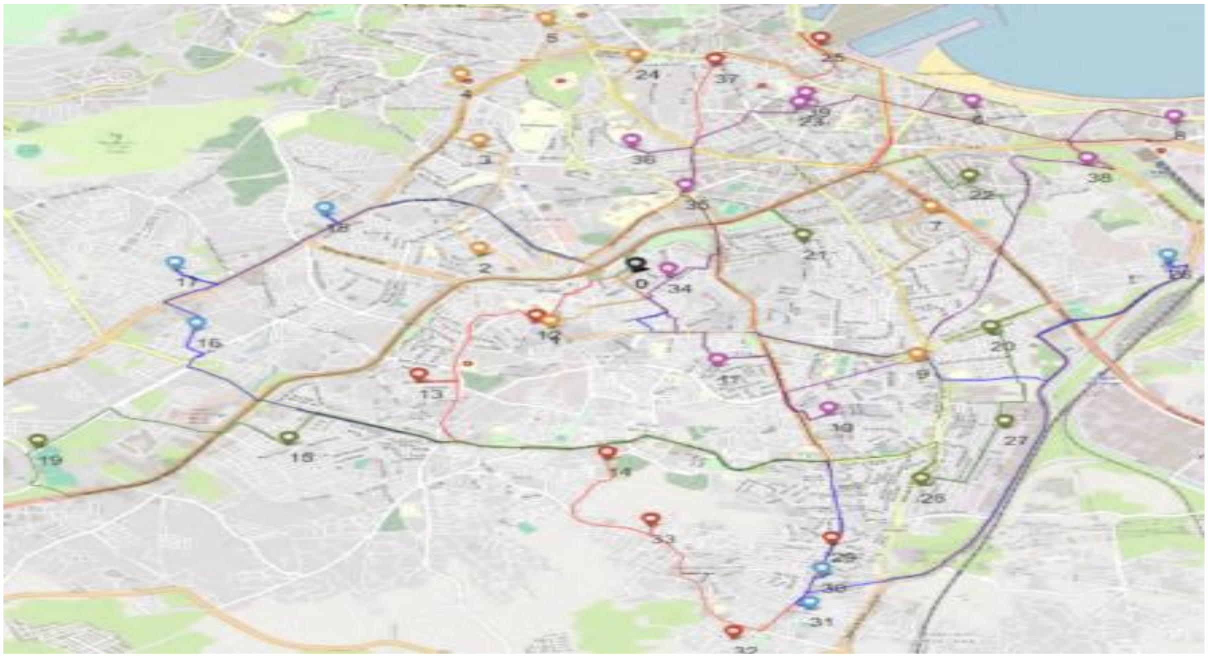


Figure 7. Comparison between PPO and ACO optimization: (**a**) PPO-Optimized Route for VRP in Tanger.

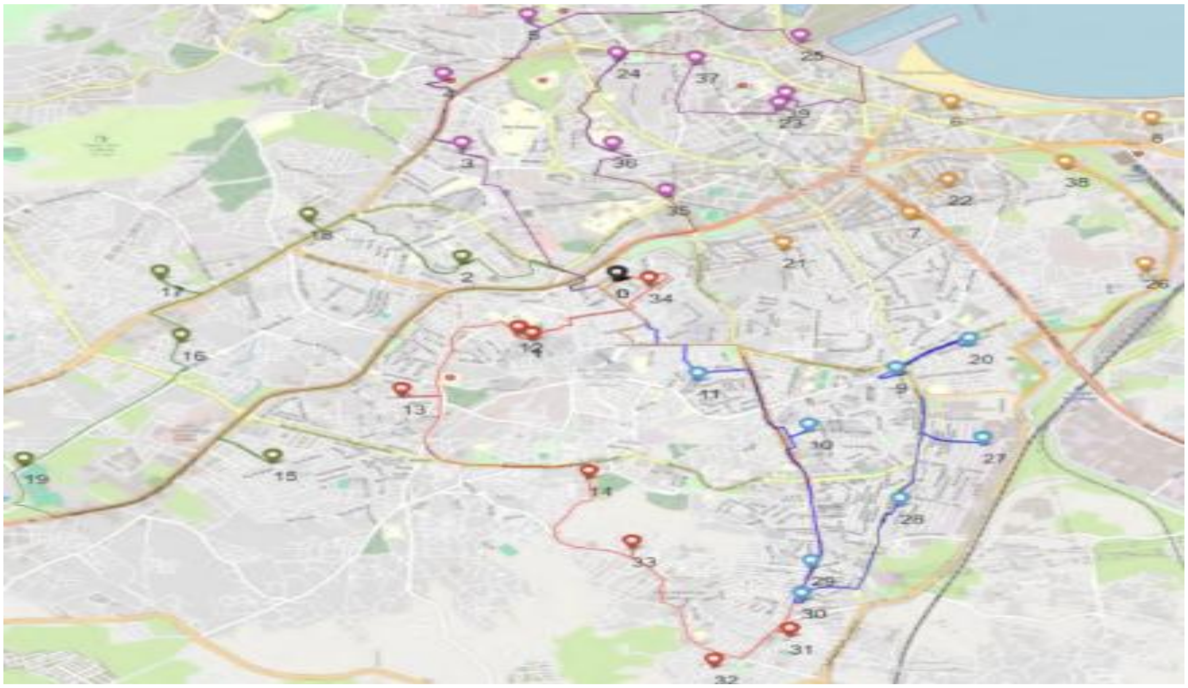


Figure 7. Comparison between PPO and ACO optimization: (**b**) ACO-ACO-Optimized Route for VRP in Tanger.

5. Discussions and Future Research

This study compared two state-of-the-art heuristics for the Vehicle Routing Problem, Ant Colony Optimization (ACO) and Proximal Policy Optimization (PPO), on three realistic last-mile datasets that vary in demand scale and fleet size. Results demonstrate that a well-trained PPO policy delivers 1 – 17 % shorter routes than an equivalently tuned ACO metaheuristic, with the largest savings observed on the densest instance (Casablanca). The improvement stems from PPO's capacity to learn joint customer-to-vehicle assignments and intra-route ordering directly from reward feedback, effectively performing a meta-optimization that encodes problem structure in its neural parameters [12].

Yet the distance advantage comes at a cost: when the time needed to train a fresh model is included, PPO's wall-clock runtime is roughly double that of ACO. In an operational setting where a single model can be reused across many daily dispatches, this training overhead is amortised and inference latency drops below one second [18]. Where every instance is unique or data are limited, ACO remains attractive because it requires no training phase, scales gracefully to new constraints, and supports straightforward parallelisation [13].

The key managerial insight is therefore contingent. Firms with high-frequency or real-time routing needs, and access to sufficient historical data, gain most from an offline-trained PPO service. Operators facing sporadic, heterogeneous problems can rely on ACO to secure near-optimal tours with minimal set-up effort. Hybridising the methods may remove this trade-off: PPO can generate a high-quality initial solution in milliseconds that ACO then refines within a short time budget, combining learning speed with swarm-based polishing [16].

Future research should first focus on blending learning and swarm intelligence rather than treating them as competing paradigms. A natural next step is a two-stage pipeline in which a pretrained PPO policy produces an initial set of routes in milliseconds, after which a short ACO or 3-opt phase refines those tours within a tight time budget [16]. Such hybrids promise the best of both worlds: the speed and data-driven insight of reinforcement learning combined with the polishing power and interpretability of pheromone-based search. Progress also depends on richer neural encoders: replacing the current shallow CNN with graph-attention or transformer layers could capture long-range spatial correlations and scale the learned policy to fleets that serve hundreds or even thousands of customers [10, 17]. Curriculum schedules and meta-learning routines will be required so the same agent can generalise from village-scale to megacity-scale instances without catastrophic forgetting.

A second line of work aims at true real-time adaptability and autonomy. Online fine-tuning of the PPO policy, using replay buffers and lightweight gradient steps during deployment, would allow the agent to react to traffic disruptions or late customer requests while preserving its prior knowledge [12]. In parallel, self-adaptive ACO variants could learn their own pheromone parameters through a lightweight reinforcement signal, removing today's manual calibration effort [13]. To build practitioner trust, both strands need explainability tools: attention heat-maps over street networks, saliency traces that reveal which customers influenced a decision, and visualisations of evolving pheromone trails can help dispatchers understand and validate automated recommendations.

Finally, secure and predictive logistics will become critical as last-mile networks grow. Our laboratory prototype already anchors VRP plans on a permissioned blockchain, creating tamper-proof route and delivery logs for regulators and supply-chain partners [8]. The next step is to pair that immutable ledger with short-term demand forecasts so the optimiser starts from an informed view of tomorrow's parcel volumes and customer geography [33]. Integrating forecasting modules directly into the optimisation loop could let the system pre-train on synthetic scenarios, warm-start daily runs, and continually improve as new data arrive, closing the gap between planning and execution in smart-locker delivery networks.

6. Conclusion

This research demonstrates that Proximal Policy Optimization (PPO) offers a transformative approach to the Vehicle Routing Problem (VRP) in smart-locker logistics. Our comparison shows that PPO consistently achieves better routing efficiency, cutting distances by up to 17% in dense urban environments like Casablanca, while Ant Colony Optimization (ACO) continues to be a reliable, training-free solution for static and sporadic tasks. PPO's near-instantaneous inference makes it perfect for high-frequency, real-time operations where the computational overhead is rapidly amortized, despite the fact that it requires substantial initial training.

The study notes that learning-based optimization is becoming increasingly important, but it also stresses that hybridity is the way of the future for logistics. Businesses can create quick, excellent delivery plans by combining PPO's predictive speed with ACO's refinement capabilities. Scaling these autonomous systems into robust, transparent, and megacity-ready delivery networks will require the integration of graph-attention networks and blockchain-secured demand forecasting in the future.

Supplementary Materials: Not applicable.

Author Contributions: All authors—Mohammed-Ali Ejjanfiand Jamal Benhra—contributed equally to the conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft preparation, writing—review and editing, visualization, supervision, project administration, and funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: This work was conducted as part of the MILEX project and greatly benefited from the unwavering support of the Hassan II University of Casablanca, Ecole Centrale Casablanca, and Mohammed VI Polytechnic University. The collaborative and constructive involvement of these institutions played a pivotal role in the successful realization of this work, and we express our profound grati-tude for their invaluable contributions throughout the research process. Thanks are also due to the refer-ees for their valuable comments.

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

Abbreviations

The following abbreviations are used in this manuscript:

ACO	Ant Colony Optimization
PPO	Proximal Policy Optimization
VRP	Vehicle Routing Problem
MDP	Markov Decision Process
RL	Reinforcement Learning

References

[1]. Li, Y., Zhang, T., & Wang, P. (2025). A deep reinforcement learning approach for vehicle routing problems. *Expert Systems with Applications, 229*, 121567.

[2]. Fan, L., Wu, X., Wang, Y., Li, Z., & Chen, Q. (2024). A two-stage hybrid ant colony algorithm for multi-depot half-open time-dependent electric vehicle routing problem. *Complex & Intelligent Systems, 10*, 2107–2128.

[3]. Sun, Q., Li, J., & Chen, W. (2023). A reinforcement learning framework for dynamic vehicle routing under stochastic demand. *Transportation Research Part E: Logistics and Transportation Review, 175*, 103068.

[4]. Wang, Z., Liu, Y., Zhao, H., Zhang, X., & Chen, Y. (2022). Improved ant colony algorithm for the split delivery vehicle routing problem. *Applied Sciences, 12*, 5090.

[5]. Lee, D., Kim, K., & Park, J. (2023). A deep reinforcement learning-based decision-making approach for routing problems. *Applied Sciences, 15*, 4951.

[6]. Patel, R., Gupta, S., & Singh, M. (2024). A comparison of reinforcement learning policies for dynamic vehicle routing problems. *Transportation Research Part E: Logistics and Transportation Review, 182*, 103308.

[7]. Ejjanfi, M.-A., & Benhra, J. (2024). Flexible routing and optimization for the last mile: An approach based on smart lockers and heuristic techniques. In *Proceedings of OLA 2024*. Retrieved May 15, 2025, from https://ola2024.sciencesconf.org/516413

[8]. Nadime, K. L., Benhra, J., Benabbou, R., & Mouatassim, S. (2023). Automating attended home deliveries with smart contracts: A blockchain-based solution for e-commerce logistics. *E3S Web of Conferences, 469*, 00026.

[9]. Foa, S., Coppola, C., Grani, G., & Palagi, L. (2022). *Solving the vehicle routing problem with deep reinforcement learning* (arXiv preprint).

[10]. Wu, X., Xiao, Y., Wu, C., Wu, Y., Yu, C., Zhou, Y., & Wang, D. (2024). *Neural combinatorial optimization algorithms for solving vehicle routing problems: A survey* (arXiv preprint).

[11]. Schulman, J., Moritz, P., Levine, S., Jordan, M. I., & Abbeel, P. (2016). *High-dimensional continuous control using generalized advantage estimation* (arXiv preprint).

[12]. Ara, S., Oion, S. R. M., Akib, M. M. K. M., Shohel, H. R. M., Ridoy, N. F. M., Kabita, F. A., & Shqhiduzzaman, M. (2023). Vehicle routing problem solving using reinforcement learning. In *Proceedings of the 26th International Conference on Computer and Information Technology (ICCIT)* (pp. 1–6). Cox's Bazar, Bangladesh.

[13]. Wwinata, W. (n.d.). *Ant colony optimization (ACO) for vehicle routing optimization (VRO)*. Retrieved May 15, 2025, from https://waresix.engineering/ant-colony-optimization-aco-for-vehicle-routing-optimization-vro-c9c8ff3d38a0

[14]. Liu, Y., Wang, Z., & Liu, J. (2024). A quick pheromone matrix adaptation ant colony optimization for dynamic customers in the vehicle routing problem. *Journal of Marine Science and Engineering, 12*, 1167.

[15]. Zhang, S., Liu, Y., & Zhao, H. (2025). An integrated optimization approach for crowd-shipping leveraging smart lockers. *Omega, 121*, 103841.

[16]. Chen, Y., Chen, M., Yu, F., Lin, H., & Yi, W. (2024). An improved ant colony algorithm with deep reinforcement learning for the robust multiobjective AGV routing problem in assembly workshops. *Applied Sciences, 14*, 7135.

[17]. Verbakel, J. (2024). *A proximal policy optimization algorithm for solving logistical optimization problems* (Master's thesis). Tilburg University, Tilburg, The Netherlands.

[18]. Iklassov, Z., Sobirov, I., Solozabal, R., & Takáč, M. (2023). Reinforcement learning for solving stochastic vehicle routing problem. In *Proceedings of the 15th Asian Conference on Machine Learning (ACML)* (pp. 502–517). İstanbul, Turkey.

[19]. Brown, T., Zhao, X., & Chen, L. (2024). Knowledge-driven ant colony optimization algorithm for instant delivery peaks. *Applied Soft Computing, 137*, 110225.

[20]. Zhao, Y., Wu, X., Xiao, Y., Yu, C., Zhou, Y., Wang, D., & Wu, C. (2025). Deep reinforcement learning for the dynamic and uncertain vehicle routing problem. *IEEE Transactions on Intelligent Transportation Systems, 26*, 3897–3911.

[21]. Silva, R., Gomes, J., & Rodrigues, A. (2024). Optimizing a dynamic vehicle routing problem with deep reinforcement learning: Analysing state-space components. *Machines, 12*, 96.

[22]. Ahmad, I., Khan, S., Ali, M., Zhang, T., & Wang, P. (2025). Intelligent deep learning-based classification with vehicle routing for balanced GHG emissions. *Procedia Computer Science, 230*, 1234–1245.

[23]. Müller, F., Kessler, R., & Schneider, M. (2025). Optimal location and capacity planning for parcel lockers. *Transportation Research Record*.

[24]. Zhao, L., Cheng, G., & Huang, Y. (2025). Reinforcement learning-based algorithm for dynamic multi-depot crowdsourced delivery. *Expert Systems with Applications, 234*, 120556.

[25]. Garcia, A., Lopez, M., Wang, X., & Cui, Y. (2025). Learning-guided bi-objective evolutionary optimization for green waste-collection routing. *Journal of Cleaner Production, 425*, 139847.

[26]. Kim, H., Lee, J., & Park, S. (2024). Designing a sustainable delivery network with parcel locker systems. *Computers & Operations Research, 160*, 106375.

[27]. Patterson, D., Thomas, A., & Chen, D. (2024). Amazon Locker capacity management. *Interfaces, 54*, 455–470.

[28]. Zhou, Q., Li, H., & Wang, F. (2025). Deep reinforcement learning-based low-energy-consumption scheduling for electric logistics vehicles. *Scientific Reports, 15*, 4820.

[29]. Ortiz, P., Racheva, Z., & Nikolova, V. (2024). Challenges and opportunities in crowdsourced delivery planning. *Annals of Operations Research, 325*, 123–147.

[30]. Lopez, M., Wang, X., & Cui, Y. (2025). Joint vehicle and courier routing problem in last-mile delivery with parcel lockers. *Expert Systems with Applications, 242*, 120786.

[31]. Tobias, V., & Jozefowicz, J. (2024). Recourse strategies for the routing problem of mobile parcel lockers. *European Journal of Operational Research, 315*, 889–905.

[32]. Nadime, K. L., Benhra, J., Benabbou, R., Mouatassim, S., Chen, D., & Patel, R. (2025). The impact of emerging blockchain trends on supply chain management. *Operations Research Forum, 6*, 24.

[33]. Marzak, Z., Benhra, J., Benabbou, R., & Mouatassim, S. (2023). Forecasting seasonal and trend-driven data: A comparative analysis of classical techniques. *Journal of Industrial Engineering, 2057*, 49–62.