International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(19s) 2026 pp. 470-491
https://doi.org/10.70917/ijcisim-2026-4608
Article

AI/ML-Driven Graph Transformer and Reinforcement Learning for Last-Mile Logistics Optimization

Ramesh Nalluri[1], Sujit Singh[2], Venkata Reddy Muppani[3]*

[1] School of Business, Woxsen University, Hyderabad – 500033, Telangana, India.
Email: ramesh.nalluri_phd.2022@woxsen.edu.in
ORCID: 0009-0005-9151-1774
[2] Professor & Research Supervisor, School of Business, Woxsen University, Hyderabad, Telangana, India.
Email: sujit.singh@woxsen.edu.in
ORCID: 0000-0002-3643-5183
[3] The Labour Relations Institute Hyderabad (TLRI Hyderabad), Hyderabad, Telangana – 500094, India.
Email: muppani@iitb.ac.in
ORCID: 0000-0002-7691-3741
Corresponding Author: Sujit Singh

Abstract: Last-mile delivery represents up to 53% of total shipping costs, yet effective route prediction tools are still hard to find. A major issue in logistics is that couriers often do not follow the best routes. Local knowledge, time constraints, and human judgment create an average Kendall Rank Correlation (KRC) of only 0.60 between the best calculated routes and actual behaviour. Bridging this gap is crucial for improving delivery time estimates, dispatching, and planning.

This research presents a Hybrid Graph Transformer-Reinforcement Learning (HGTRL) framework, a new hybrid deep learning model aimed at predicting real-world courier pickup sequences under spatial and temporal limits. Instead of assuming perfect behaviour, HGTRL learns from historical courier choices using the large LaDe dataset. The model structure includes a two-layer, eight-head Transformer encoder and a Long Short-Term Memory (LSTM) attention decoder. To effectively capture complex routing preferences, HGTRL uses a hybrid loss function that combines Maximum Likelihood Estimation (MLE) with the REINFORCE algorithm ($L_{hybrid} = L_{MLE} + \lambda L_{RL}$, $\lambda=0.3$). This approach allows the model to imitate real behaviour while improving route sequences. Extensive evaluation covered 10.67 million packages in three diverse cities: Chongqing, Shanghai, and Yantai.

While earlier models like DeepRoute and Graph2Route showed the benefits of learning from historical data, HGTRL sets a new benchmark for performance. Compared to the leading baseline, DRL4Route2, HGTRL made significant gains across all key metrics: KRC improved by 14.32 to 15.50 points, Location Square Deviation (LSD) dropped by 0.11 to 0.50, and Edit Distance (ED) went down by 0.17 to 0.32. Additionally, ablation studies showed that removing the reinforcement learning element alone led to a 3.2% decrease in KRC, stressing its importance. Ultimately, HGTRL's precise sequence predictions allow for reliable estimated arrival times and adaptive order assignments, greatly boosting fleet efficiency and customer satisfaction in today's logistics systems.

Keywords: Route prediction, Last-mile, logistics, Transformer networks, Reinforcement learning and Spatial-temporal optimization.

1. Introduction

The rapid growth of e-commerce and on-demand delivery has significantly changed how goods move through cities. It has also created unique challenges for last-mile package deliveries and first-mile pickup operations. The

Copyright: © 2026 by the authors

global e-commerce affluent has put last-mile delivery at the centre of supply chain research and industry investment. Last-mile logistics make up 41 to 53% of total supply chain costs. It is also the most labour-intensive part of urban freight distribution [1, 2]. In China, more than 10 billion packages are delivered each month, especially in megacities like Shanghai, where daily parcel volumes exceed several million units. The courier workforce plays an essential role, and being able to compute near-optimal delivery routes in real time is crucial for both economic and social reasons.

At its core, courier route planning is a version of the Vehicle Routing Problem (VRP). This is a type of combinatorial optimization problem known to be NP-hard [3]. For 100 delivery stops, the solutions equal 100! permutations, making exact solutions impractical within operational time limits. Traditional methods, like the Clarke-Wright savings algorithm and Lin-Kernighan local search, can provide polynomial-time approximations. However, they do not capture the implicit knowledge that experienced human couriers have. This knowledge includes traffic patterns, building access constraints, recipient availability, and informal micro-routing practices that are hard to define formally [4].

While early deep learning models trained solely on historical sequences lose accuracy beyond 20 stops and are unable to optimize for the route quality metrics that actually drive operational outcomes, existing route prediction methods continue to fall short despite numerous scale and complexity issues. Classical optimization tools assume couriers follow mathematically ideal paths. This unresolved contradiction served as the impetus for this study: the requirement for a model that simultaneously guides the creation of better routes rather than merely replicating previous ones and learns how couriers actually behave, not how they should in theory.

Two theories comprise the current deep learning methods for route prediction. Pointer Networks [4] and Graph2Route [5] are two examples of imitation learning techniques that obtain good next-stop accuracy through teacher-forced MLE but suffer from exposure bias: prediction errors compound at following steps, lowering full-route quality. On the other hand, without st0rong imitation priors, reinforcement learning frameworks like DRL4Route [6] optimize sequence-level rewards but show delayed convergence and significant gradient variance.

At the junction of both paradigms, there remains a crucial and unexplored gap: no previous study has systematically merged an LSTM pointer decoder trained under a composite MLE-RL objective that directly maximizes rank-order fidelity with a Transformer-based global encoder. As a result, there is significant space for improvement in the global ordering quality that defines full-day qdelivery efficiency, with state-of-the-art models achieving Kendall Rank Correlation scores of roughly 57%. This study aims to investigate the following questions:

1. Does replacing a GCN encoder with a multi-head Transformer encoder yield statistically significant improvements in global route ordering (KRC) and positional accuracy (LSD) over the DRL4Route baseline?
2. To what extent does a composite MLE + REINFORCE training objective outperform either paradigm in isolation across diverse urban geographies?
3. How do city-specific density characteristics -Shanghai (flat, high-density), Chongqing, Yantai (coastal, medium-density) -modulate model performance across the proposed metric suite?

Instead of the difference between delivery drivers' theoretically optimal routes and what they actually do, how delivery drivers operate can be critical to the success of downstream delivery operations, such as predicting when items will be delivered [7] and dispatching orders [8]. Because of the disparity between the theoretically best curves for delivery drivers and the actual ways in which these drivers operate, real deliveries may be a critical factor in accurately estimating things like the expected time of delivery for an item [7] or dispatching orders to delivery persons [8] to deliver items.

The concept of "perceived distance" in route prediction was first introduced by Zhang, Liu [8], who discovered that couriers' spatial abilities are influenced by factors other than the actual distance to their destinations. Their OSquare technology, which was deployed in more than 2000 cities and on an Ele.me platform with 168 million users, was demonstrated to have an influence on overdue delivery rates of 48.02% due to accurate route prediction, indicating that this line of study has significant practical value.

Building on this research foundation, Wen, Lin [7] constructed DeepRoute, the first deep neural network created especially for package pick-up path prediction since its launch in 2021. They analyzed past data from 432 couriers who had fulfilled more than 2.32 million delivery requests using transformer encoders. With Graph2Route at KDD 2022, Wen, Lin [5] later developments moved from sequence-based to graph-based representations, explicitly modeling package locations as nodes in a dynamic spatial-temporal graph. The LaDe dataset, the first extensive, large-scale, publicly accessible dataset for last-mile logistics research, was most recently published by Wu, Wen [9]. It includes 10,677k shipments from 21k couriers over a period of six months in several Chinese cities.

The four principal contributions of this study to the field of last-mile courier route prediction are:

i. Novel Hybrid Architecture-

In order to overcome the drawbacks of both solely supervised and reinforcement learning approaches, research suggests a novel hybrid architecture. The model combines an LSTM-based pointer network decoder that automatically produces pickup sequences with an 8-head, 2-layer Transformer encoder that captures long-range spatial dependencies across all delivery nodes. These two elements are connected by a carefully calibrated hybrid loss function, L_hybrid = L_MLE + 0.3·L_RL, which concurrently optimizes toward route quality metrics and learns from ground-truth courier behavior. Ablation experiments reveal that eliminating the reinforcement learning component alone results in a 3.2% decrease in Kendall Rank Correlation, proving that the RL objective is a structural requirement rather than an optional improvement, demonstrating the significance of this design decision.

ii. New State-of-the-Art Performance

HGTRL outperforms the previous state-of-the-art (PAPN at 69.3%) and improves global ranking alignment by 14.32% to 15.50% over the previous leading baseline, DRL4Route2, with a peak Kendall Rank Correlation (KRC) of 72.70%. Additionally, it dramatically lowers Edit Distance (ED) by 0.17 to 0.32 and Location Square Deviation (LSD) by 0.11 to 0.50. Despite these significant worldwide advancements, HR@3 is still quite competitive in all three cities. This implies that rather than favoring one statistic over another, the hybrid framework strengthens the model overall. These improvements are fueled by a joint MLE-RL loss function in which the REINFORCE reward directly optimizes non-differentiable route quality indicators by integrating KRC and LSD.

iii. Reproducible Mathematical Framework Complete mathematical formulations are provided for every component of the architecture, from the multi-head self-attention mechanism and glimpse-based decoder to the combined reward signal and hybrid loss computation. Alongside these, fully worked pseudo-code is included for both the training procedure.

iv. Large-Scale Multi-City Empirical Validation Validated the model across 10.67 million packages in three geographically distinct Chinese cities -dense urban Shanghai, mountainous Chongqing, and coastal mid-density Yantai -demonstrating robustness across varied urban environments and contributing new insight into how urban characteristics interact with courier behaviour predictability.

Section 2 of this paper presents the literature review; Section 3 Specifies the Research Gap and presents the HGTRL architecture. Section 4 Describes the experimental setup. Section 5 reports empirical results. Section 6 discusses implications and limitations. Section 7 concludes the paper.

2. Literature Review

Using the keywords "last mile," "machine learning," "artificial intelligence," "machine learning in last mile in ecommerce," "artificial intelligence in last mile," and "machine learning in last mile," we were able to find pertinent literature. Academic databases, including Scopus, IEEE, ScienceDirect, Taylor & Francis, and Business Source from the internet, were employed to guarantee thoroughness. A total of 5,569 papers were screened from Springer (n = 5,135), IEEE (n = 46), ScienceDirect (n = 66), and Taylor & Francis (n = 322). The screening technique produced 276 research articles after eliminating duplicates and publications outside of its focus; the previous eligibility procedure produced 81 articles. Figure 1 illustrates the research lineage and citation network connecting the four key papers that

form the foundation of this work. This network shows the evolution from classical optimisation approaches to supervised learning, then to graph-based methods, culminating in the hybrid framework presented in this paper.

Figure 1: Research Citation Network and Evolution Timeline

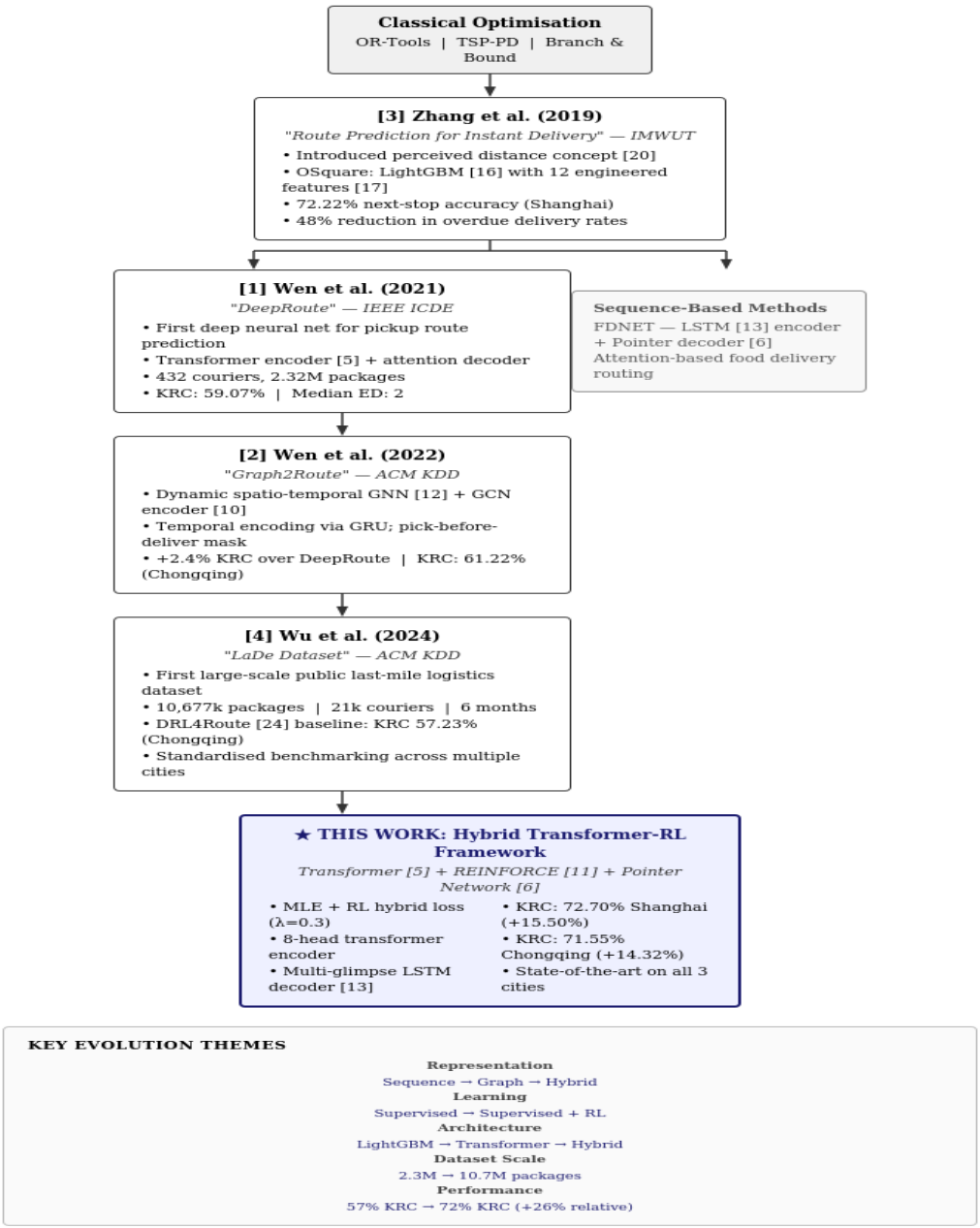

Figure 1: Research Citation Network and Evolution Timeline

Keyword co-occurrence analysis was performed using VOSviewer software. Just nine of the 153 keywords satisfied the required minimum of four occurrences. Last-mile delivery, vehicle routing, city logistics, digital transformation, logistics, optimization, routing, and transportation with machine learning were the themes that emerged.

Table 1: Keywords Occurrences and Link Strength (Author own depiction)

ID	Keyword	Occurrences	Total Link Strength
151	last-mile delivery	11	51
148	last mile delivery	6	29
290	vehicle routing	5	17
51	city logistics	4	17
88	digital transformation	4	19
164	Logistics	4	18
190	Optimization	4	26
229	Routing	4	24
270	Transportation	4	25
61	covid-19	3	16
98	drone delivery	3	13
102	Drones	3	22
112	electronic commerce	3	22

2.1 Route Prediction Methodologies

Classical route optimization in logistics is based on the Vehicle Routing Problem (VRP) [10] and its many variations. The Traveling Salesman Problem with Pickups and Deliveries (TSPPD) Gendreau, Laporte and Vigo [11] has more constraints than the classic TSP since it requires products to be picked up prior to delivery, creating priority constraints. In a large empirical study using Ele.me data, Zhang, Liu [8] demonstrated that couriers rarely use the distance-optimal route, with an average Kendall rank correlation coefficient of only 0.60. However, branch and bound algorithms and Google's OR-Tools package are good methods for solving moderately sized problems. The 16.5% prediction error between the ideal and actual route is caused by couriers having a better understanding of the traffic patterns in their area or being constrained by a time window that requires complicated trade-off results, as well as psychological aspects of the decision-making process when choosing routes that go beyond distance travelled.

2.2 Deep Learning and Sequence Modelling in Courier Route Prediction

Route prediction was initially accomplished in OSquare [8] by converting the LightGBM [12] route prediction approach into a framework for estimating the next location based on features that were accurately designed utilizing 12 different criteria. Perceived distances (D_sd and D_other), time urgency (T_ToD and T_budget), and spatial linkages (B_nearest and N_sameType) between consecutive delivery were among them. By using chi-squared (χ^2) testing and variance analysis to pick those traits, they were able to predict the next stop on Shanghai route orders with an average accuracy rate of 72.22% [13].

This work was improved by DeepRoute [7], which used a transformer-based model in place of the LightGBM model and added two Encoder Heads, each with eight heads. They accordingly achieved a KRC of 59.07%. After more than 20 stops, accuracy often decreases by 5% to 7% due to sequence-based architectures' essential constraints in modelling the spatial relationship between packages and the challenge of effectively modeling large time series of events.

2.3 Graph-Based and Reinforcement Learning Approaches:

Graph2Route [5] used GCN encoders [14] with edge gates ($G_{ij} = \sigma(\tilde{E}_{ij})$) and node updates utilizing message passing over k-nearest neighbors (usually k=n-1) to generate dynamic spatial-temporal graph neural networks [15]. Using four restrictions (completed, pick-before-deliver, and neighbor limitations), the mask technique that eliminated

impractical nodes significantly reduced the search space, achieving 61.22% KRC on Chongqing data (a 2.4% improvement over DeepRoute).

2.4 Large-Scale Benchmark Datasets and Reproducible Last-Mile Logistics Research

The LaDe dataset [9], a ground-breaking step toward verifiable research, includes: (1) 10,677k packages sent over a six-month period by 21k couriers; (2) complete package information along with the original task event logs and complete trajectory of each delivery person; and (3) data created from various scenarios, including pick-only (Shanghai logistics) with 1.45 million packages dispatched by 4,502 different couriers. In order to facilitate thorough empirical method comparisons, graph structured instances of V, edge features E, and reachability masks M can be determined using the consistently pre-processed LaDe dataset. The methodology is applied to provide a foundation for future research advancements with respect to the state-of-the-art DRL4Route, which achieved an overall KRC of 57.23% at Chongqing.

3. Problem Formulation and Mathematical Preliminaries

Predicting the next sequence in a set of spatial-temporal and graph-based data is how research defines the problem of forecasting a courier's itinerary. This section contains baseline mathematical definitions that are necessary for this work as well as formal definitions based on sighting conventions.

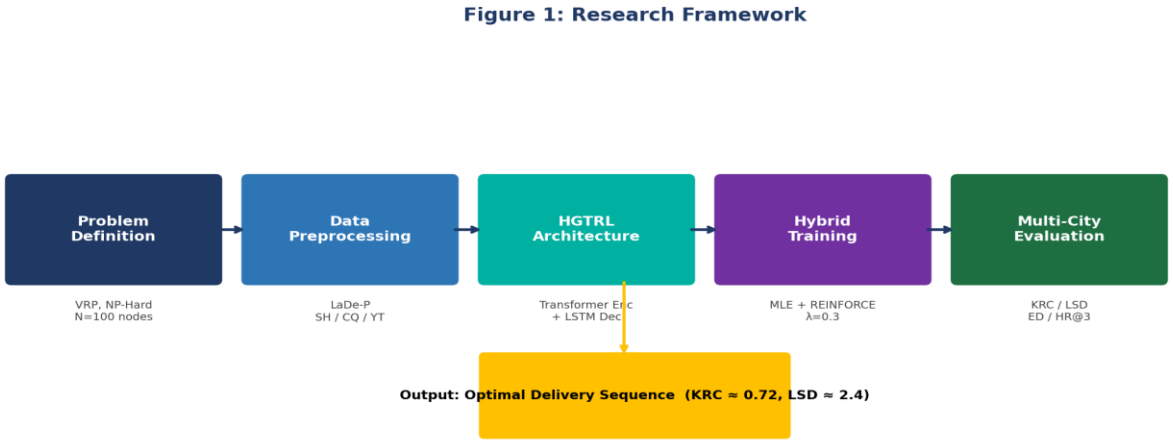

Figure 2. Research Framework- Conceptual flow (Authors Own depiction)

3.1 Problem Formulation

The input representation, which can comprise a set of incomplete jobs given the status of the courier at the current instantaneous time t, when the courier has been assigned and is on its way to complete the jobs, consists of:

• Node Features:

Encode per-packet attributes (latitude, longitude, time window starts, time window end, service time, volume, and two auxiliary context features) across a batch where B is the number of batches, T is the number of trajectories, N ≤ 100 delivery nodes, $\boldsymbol{d_v = 8}$ node feature dimensions.

• Edge Features:

$$\boldsymbol{E} \in \mathbb{R}^{\boldsymbol{B}\times\boldsymbol{T}\times\boldsymbol{N}\times\boldsymbol{N}\times\boldsymbol{d_e}}$$

Encode pairwise spatial and temporal distances where $\boldsymbol{d_e = 4}$ edge feature dimensions.

• The Binary Reachability Mask:

$$\boldsymbol{M} \in \{\mathbf{0},\mathbf{1}\}^{\boldsymbol{B}\times\boldsymbol{T}\times\boldsymbol{N}}$$

Indicating the valid nodes unvisited nodes for the time frame of each trajectory timestep.

• Expected Result:

Route sequence $\boldsymbol{\pi} = (\boldsymbol{\pi_1}, \boldsymbol{\pi_2}, \ldots, \boldsymbol{\pi_N})$, where $\boldsymbol{\pi_i} \in \{\mathbf{1}, \ldots, \boldsymbol{N}\}$.

The objective is to learn a parameterised policy that predicts the delivery sequence π = (π₁, π₂, …, π_N) such that the predicted sequence maximally correlates with the ground-truth expert sequence π* under the Kendall Rank Correlation and Levenstein Sequence Distance metrics.

3.2 Model Architecture - Route Constraints and Feasibility

1. A real-world route prediction must adhere to a number of constraints:

(1) Each destination may only be visited once.
(2) Each order must have a pick-up prior to delivery.
(3) The time frame for promised delivery must be followed.
(4) concurrent orders are limited by available capacity. These constraints are recorded in the reachability mask M_t at all times.

INPUT LAYER
Delivery Node Set {s1, s2, ..., sN} with feature vectors: [latitude, longitude, time_window_start, time_window_end, volume, elapsed_time]
TRANSFORMER ENCODER
2 Stacked Multi-Head Self-Attention Layers | H = 8 Attention Heads | d_model = 64 | Feed-Forward Dimension = 256 Attention(Q,K,V) = softmax(QK^T / √d_k) · V → Global Context Embeddings {h1, h2, ..., hN}
ATTENTION-LSTM DECODER
LSTM Hidden State ht | Multi-Glimpse Pointer Mechanism (K=3) | Context Vector ct from Encoder Embeddings Pointer Score: e(j,t) = v^T · tanh(W1·hj + W2·ht) | Softmax over unmasked nodes
REACHABILITY MASK
Dynamically masks visited nodes | Enforces: no repeat visits, pickup-before-delivery precedence, capacity limits Feasible Set F(t) shrinks at each decoding step t → Route validity guaranteed
HYBRID LOSS FUNCTION
L_hybrid = L_MLE + λ·L_RL (λ=0.3) L_MLE: Cross-entropy with teacher forcing | L_RL: REINFORCE with beam-search baseline (k=5) Reward R = ΔLSD + 6.0·ΔKRC + w_HR·ΔHR
OUTPUT
Predicted Delivery Sequence π = [π1, π2, ..., πN] Optimised jointly for: KRC (↑ global ordering) + LSD (↓ positional errors) + HR@3 (↑ local accuracy)

Figure 3: HGTRL Model Architecture (Authors Own depiction)
-Transformer Encoder + Pointer-LSTM Decoder with Hybrid MLE-RL Training

3.2.1 Architectural Component Data Flow

To fully contextualize the highly complex spatial-temporal data flow represented in the architectural schematic, the network is fundamentally decomposed into six rigorously defined, mathematically distinct operational components. Each component is engineered to resolve a specific bottleneck in combinatorial route optimization.

1. Input Layer: Translating Logistics Reality into High-Dimensional Tensors

The Input Layer serves as the foundational interface, responsible for translating the chaotic, heterogeneous physical reality of the urban logistics network into standardized, mathematically digestible high-dimensional tensors. This translation is vital; neural networks cannot process raw maps, but they excel at processing structured matrix embeddings.

- **Delivery Node Set $\{s_1, s_2, \dots, s_N\}$:** This set mathematically represents the entire universe of available, uncompleted package deliveries assigned to a specific courier at an instantaneous observation time t. To maintain strict computational efficiency and bound the exponential explosion of the algorithmic search space, the model architecture imposes a hard mathematical maximum constraint of $N \leq 100$ individual nodes per assigned route.
- **Feature Vectors ($V \in \mathbb{R}^{B \times T \times N \times d_v}$):** Rather than treating a delivery simply as a dot on a map, each node s_i is intricately parameterized by a dense, continuous feature vector $d_v = 8$, encompassing critical environmental and temporal states:
    - **Latitude & Longitude:** These represent the normalized, absolute GPS coordinates, defining precise spatial positioning and establishing the core physical geometry of the routing graph.
    - **Time_Window_Start & Time_Window_End:** These two dimensions strictly encode the Service Level Agreement (SLA) boundaries dictated by the purchasing customer. For example, if a high-priority package must be delivered strictly between 14:00 and 16:00, the network must computationally weigh this temporal urgency against pure spatial proximity to prevent SLA violations.
    - **Volume:** This parameter encodes the physical cubic capacity that the specific package occupies. In real-world logistics, couriers operate vehicles (e.g., electric tricycles or delivery vans) with strict volumetric limits. Feeding this to the network is critical for enforcing hard vehicle load capacity constraints during the generative decoding phase.
    - **Elapsed_Time:** A highly dynamic, continuously updating temporal feature representing exactly how long the courier has been operating on the current shift. This allows the sophisticated attention mechanisms to implicitly account for human operational factors, such as driver fatigue or the impending urgency to return to the depot as the shift-end approaches.

2. Transformer Encoder: Achieving Infinite-Range Spatial Dependency Capture

The Transformer Encoder represents the most aggressive architectural departure from contemporary logistics models. It explicitly replaces traditional, highly localized Graph Convolutional Networks (GCNs) in order to capture infinite-range spatial relationships simultaneously across the entire geographic node set.

- **Dimensionality and Projection Space:** Before attention can be calculated, the raw 8-dimensional input features are linearly projected into a highly parameterized latent space where the core embedding dimension is expanded to $d_{model} = 64$. To ensure the network has sufficient capacity to learn non-linear feature interactions, the internal position-wise feed-forward networks (FFN) utilize an expanded hidden dimension of 256.
- **Stacked Multi-Head Architecture:** The encoder achieves its deep structural understanding by stacking **2 Multi-Head Self-Attention Layers**. Crucially, it utilizes **H = 8 parallel attention heads**. This multi-head design is revolutionary for routing; it allows the model to simultaneously evaluate 8 entirely distinct topographical and temporal relationships. For instance, Head 1 might focus purely on spatial geographic clustering, Head 2 might focus strictly on time-window urgency, and Head 3 might cross-reference package volumes to optimize vehicle loading patterns.

- **Mathematical Operation and Output:** The core self-attention mechanism is mathematically formulated as $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \cdot V$. By executing this scaled dot-product across all node pairs concurrently, the outputs of this layer produce the highly refined **Global Context Embeddings** $\{h_1, h_2, \ldots, h_N\}$. This effectively grants the subsequent decoder a perfect, omni-directional "bird's-eye view" of the entire delivery map, entirely immune to the information decay that plagues standard graph traversal algorithms.

3. Attention-LSTM Decoder: The Autoregressive Routing Agent

While the encoder understands the global map, the Attention-LSTM decoder acts as the active, artificial "courier," operating autoregressively to step-by-step construct the route sequence.

- **LSTM Hidden State (h_t):** The core sequential memory engine of the decoder. At every generative timestep t, the Long Short-Term Memory cell mathematically updates its internal hidden state (h_t) based on the embedding of the previously visited geographic node. This perfectly simulates the path-dependent nature of a human courier traversing a city block-by-block.
- **Multi-Glimpse Mechanism ($K = 3$):** Before committing to a final routing decision, the decoder executes a complex cognitive process simulating hesitation and verification. It performs 3 sequential mathematical "glimpses" ($K = 3$). During each glimpse, it queries the encoder's global context embeddings (c_t) against its current hidden state. This iteratively refines the internal context vector, ensuring the decoder hasn't missed critical, distant spatial clustering opportunities that a naive greedy algorithm would overlook.
- **Pointer Network Score Matrix:** The final destination selection is calculated via the pointer network equation: $e(j,t) = v^T \cdot \tanh(W_1 \cdot h_j + W_2 \cdot h_t)$. This generates a raw, unbounded logit score for every single available delivery node in the network. A final softmax activation function converts these raw logits into a normalized probability distribution, strictly bounded between 0 and 1, but critically, this softmax is evaluated *only* over the unmasked, legally available nodes.

3.2.1 Transformer Encoder

Raw node features are first projected into a $\boldsymbol{d_{\text{model}}}$ = 64-dimensional embedding space via a learnable linear transformation:

$$\boldsymbol{X^{(0)} = W_{\text{emb}} V + P}, \text{ where } \boldsymbol{W_{\text{emb}}} \in \mathbb{R}^{\boldsymbol{d_{\text{model}}} \times \boldsymbol{d_v}}$$

The embedded representation is processed through L = 2 identical Transformer layers, each comprising a multi-head self-attention sublayer followed by a position-wise feed-forward network with residual connections and layer normalisation.

Within each self-attention sublayer, H = 8 attention heads compute scaled dot-product attention in parallel. For head h:

$$\boldsymbol{Q^{(h)} = XW_Q^{(h)}, K^{(h)} = XW_K^{(h)}, V^{(h)} = XW_V^{(h)}}, \text{ where } \boldsymbol{W_Q^{(h)}, W_K^{(h)}, W_V^{(h)}} \in \mathbb{R}^{\boldsymbol{d_{\text{model}}} \times \boldsymbol{d_k}}$$

The attention matrix is:

$$\textbf{Attention}^{(h)} = \textbf{softmax}\left(\frac{\boldsymbol{Q^{(h)}(K^{(h)})^T}}{\sqrt{\boldsymbol{d_k}}} + \boldsymbol{M_{\text{attn}}}\right)\boldsymbol{V^{(h)}}$$

where $\boldsymbol{d_k}$ = 8 and M_attn masks padding nodes to $-\infty$. Multi-head outputs are concatenated:

The feed-forward sublayer applies

$$\mathbf{FFN}(x) = \mathbf{ReLU}(xW_1 + b_1)W_2 + b_2$$

with d_ff = 256. A mean-pooling graph embedding initialises the decoder state.

$$\bar{h} = \frac{1}{N}\sum_{i=1}^{N} h_i^{(L)}$$

3.2.2 Attention-Based LSTM Decoder

The decoder generates the route autoregressively. At each step t, an LSTM updates its hidden state:

$$(h_t, c_t) = \mathbf{LSTM}(x_{t-1}, (h_{t-1}, c_{t-1}))$$

where x_{t-1} is the embedding of the previously selected node. A multi-glimpse mechanism initialised as

$$g_{t,0} = h_t$$

Refines the context:

masked nodes receive

$$g_{t,k} = \sum_{i=1}^{N} \alpha_i^{(k)} h_i$$

The pointer mechanism then assigns:

where M_t is the mask at time step t (already selected nodes are masked).

The dynamic mask M_t prevents revisiting already-selected nodes.

3.3 Hybrid Training Objective

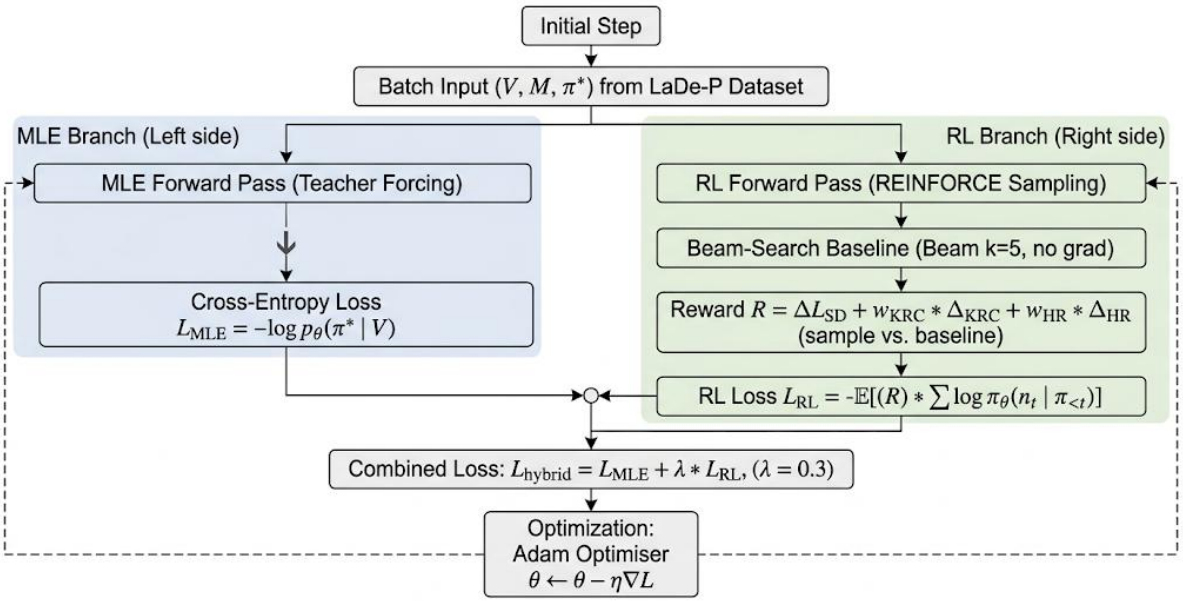

Figure 4. Hybrid Training Workflow: Authors own depiction

Following the initial batch input, the process divides into two parallel branches: Branch of MLE (Supervised Learning): "Teacher Forcing" is used by this team on the forward pass. In order to determine how well the model predicts the ground-truth sequence given the input V, it computes a standard Cross-Entropy Loss .$\pi^* \mathcal{L}_{\text{MLE}}$

Policy Optimization, or RL Branch: The REINFORCE algorithm is used on this side. To determine a relative Reward R, the model samples outputs and compares them to a "Beam-Search Baseline" (k=5). A weighted sum of several metrics makes up this reward. To promote high-reward behaviors, the RL loss (L_RL) is then calculated. The two losses are combined to create a Lastly, the Adam Optimizer balances reward-based performance with supervised accuracy by updating the model parameters based on this hybrid gradient.

3.3.1 MLE Loss (Teacher Forcing)

The MLE component minimises the negative log-likelihood of the ground-truth sequence:

$$\mathcal{L}_{\text{MLE}} = -\frac{1}{\mid D \mid} \sum_{(\pi^*, V) \in D} \sum_{t=1}^{|\pi^*|} \log p_\theta(\pi_t^* \mid \pi_{<t}^*, V)$$

At each decoding step, the decoder receives the true previous node (teacher forcing), stabilising early training and providing a strong imitation prior.

3.3.2 RL Loss (REINFORCE with Baseline)

The RL component employs REINFORCE with a beam-search baseline (k = 5) to optimise sequence-level reward. A sample trajectory π_sample is drawn from the policy, and a baseline π_beam is produced by beam search without gradient tracking.

The combined reward is:

$$R_{\text{combined}} = \underbrace{\left(R_{\text{LSD}}^{\text{greedy}} - R_{\text{LSD}}^{\text{sample}}\right)}_{\text{LSD improvement}} + w_{\text{KRC}} \underbrace{\left(R_{\text{KRC}}^{\text{sample}} - R_{\text{KRC}}^{\text{greedy}}\right)}_{\text{KRC improvement}} + w_{\text{HR}} \underbrace{\left(R_{\text{HR/Acc}}^{\text{sample}} - R_{\text{HR/Acc}}^{\text{greedy}}\right)}_{\text{HR/Acc improvement}}$$

Where $w_{\text{KRC}} = 6.0$

The RL loss is: $\mathcal{L}_{\text{RL}} = -\mathbb{E}_{\pi \sim \pi_\theta}\left[(R(\pi) - R(\pi_{\text{greedy}})) \sum_{t=1}^{T} \log \pi_\theta(\pi_t \mid \pi_{<t})\right]$

The final objective is:

$\mathcal{L}_{\text{hybrid}} = \mathcal{L}_{\text{MLE}} + \lambda \mathcal{L}_{\text{RL}}$, where λ = 0.3.

3.4 Inference: Beam Search Decoding

At inference time, a beam search with width K = 5 is performed. At each step, the decoder keeps the K highest-scoring partial routes, expands them with the top-K next-node candidates, and lops to keep the globally best K partial sequences. The path with the highest cumulative log-probability at completion is chosen as the final prediction.

4. Hybrid Transformer-RL Methodology

This hybrid architecture comprises three key components:

(1) Global context modeling from a Transformer encoder [16].
(2) Attention-driven sequences from an LSTM decoder [17] with a pointer network [18].
(3) Hybrid training objectives based on MLE and reinforcement learning [19].

The mathematical requirements for these components are listed below.

4.1 Transformer Encoder Architecture

Node features are encoded using an encoder neural network with L=2 stacked layers of a transformer network and 8 attention heads (H=8). The following equation is used to compute the embedding of input features in the transformer input space.

Input Embeddings:

$$X^{(0)} = W_{\text{emb}}V + P$$

- $W_{\text{emb}} \in \mathbb{R}^{d_{\text{model}} \times d_v}$ is the input embedding matrix
- $P \in \mathbb{R}^{N \times d_{\text{model}}}$ is positional encoding (optional in this implementation.

Multi-Head Self-Attention:

For each head h=1...,H:

$$Q^{(h)} = XW_Q^{(h)}, K^{(h)} = XW_K^{(h)}, V^{(h)} = XW_V^{(h)}$$

$$\text{Attention}^{(h)} = \text{softmax}\left(\frac{Q^{(h)}(K^{(h)})^T}{\sqrt{d_k}} + M_{\text{attn}}\right)V^{(h)}$$

$$\text{MultiHead}(X) = \text{Concat}\left(\text{Attention}^{(1)}, \dots, \text{Attention}^{(H)}\right)W_O$$

Feed-Forward Layer:

$$\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2$$

For Each Transformer Layer:

$$X' = \text{LayerNorm}\left(X + \text{MultiHead}(X)\right)$$

$$X^{(l)} = \text{LayerNorm}(X' + \text{FFN}(X'))$$

Graph-Level Representation:

$$\bar{h} = \frac{1}{N}\sum_{i=1}^{N} h_i^{(L)}$$

where $h_i^{(L)}$ is the final node representation after L transformer layers.

4.2 Attention-Based LSTM Decoder with Pointer Network

The decoder constructs the route sequence autoregressively, utilizing LSTM [13] with multi-glimpse attention and a pointer method [6] to choose nodes from the input graph.

LSTM Cell Update:

$$(h_t, c_t) = \text{LSTM}(x_{t-1}, (h_{t-1}, c_{t-1}))$$

Glimpse Mechanism (K=2):

$$u_i^{(k)} = v^T \tanh(W_q^{(k)} g_{t,k-1} + W_r^{(k)} h_i)$$

$$a_i^{(k)} = \begin{cases} -\infty & \text{if node } i \text{ is masked} \\ u_i^{(k)} & \text{otherwise} \end{cases}$$

$$\alpha^{(k)} = \text{softmax}(a^{(k)})$$

$$g_{t,k} = \sum_{i=1}^{N} \alpha_i^{(k)} h_i$$

Pointer:

$$u_i^{\text{ptr}} = v^T \tanh(W_q^{\text{ptr}} g_{t,K} + W_r^{\text{ptr}} h_i)$$

Decoding:

$$\boldsymbol{\pi_t = \arg\max_i \log p(\pi_t = i \mid \pi_{<t}, V, E)}$$

4.3 Hybrid Training Objective: MLE + Reinforcement Learning

We have developed a method for training that includes both supervised and reinforcement learning. The hybrid loss function is:

$\mathcal{L}_{\text{hybrid}} = \mathcal{L}_{\text{MLE}} + \lambda \mathcal{L}_{\text{RL}} \lambda = 0.3$

MLE Loss (Teacher Forcing):

$$\mathcal{L}_{\text{MLE}} = -\frac{1}{|D|} \sum_{(\pi^*, V) \in D} \sum_{t=1}^{|\pi^*|} \log p_\theta(\pi_t^* \mid \pi_{<t}^*, V)$$

RL Loss (REINFORCE with Baseline):

$$\mathcal{L}_{\text{RL}} = -\mathbb{E}_{\pi \sim \pi_\theta} \left[\left(R(\pi) - R(\pi_{\text{greedy}}) \right) \sum_{t=1}^{T} \log \pi_\theta(\pi_t \mid \pi_{<t}) \right]$$

Combined Reward:

$$R_{\text{combined}} = \underbrace{\left(R_{\text{LSD}}^{\text{greedy}} - R_{\text{LSD}}^{\text{sample}} \right)}_{\text{LSD improvement}} + w_{\text{KRC}} \underbrace{\left(R_{\text{KRC}}^{\text{sample}} - R_{\text{KRC}}^{\text{greedy}} \right)}_{\text{KRC improvement}} + w_{\text{HR}} \underbrace{\left(R_{\text{HR/Acc}}^{\text{sample}} - R_{\text{HR/Acc}}^{\text{greedy}} \right)}_{\text{HR/Acc improvement}}$$

The parameter weights are $w_{\text{KRC}} = 6.0$ and $w_{\text{HR}} = 1.0$ (validated through hyperparameter tuning).

Route Quality Metrics:

KRC (Kendall Rank Correlation) [8]:

$$\text{KRC}(\pi, \pi^*) = \frac{n_c - n_d}{\frac{1}{2} n(n-1)}$$

LSD (Levenshtein Square Deviation):

$$\mathrm{LSD}(\pi, \pi^*) = \min \text{edit operations to transform } \pi \text{ to } \pi^*$$

Hit Rate @k (HR @k)

$$\mathrm{HR}\ @\mathrm{k} = \frac{1}{|\ D\ |} \sum_{i=1}^{|D|} \mathbb{1}[\mathrm{rank}(\pi_{\mathrm{true}}^{(i)}) \leq k]$$

ED (Edit Distance) [23]: the Levenshtein distance between two sequences.

$$\mathrm{ED}(\pi, \pi^*) = \mathrm{LSD}(\pi, \pi^*)$$

4.4 Training Algorithm (Pseudo-code)

Algorithm 1 presents the complete training procedure:

Algorithm 1: Hybrid Training Procedure

```
Input: Training dataset D = {(V, E, M, π*)}
    Hyperparameters: λ=0.3, w_KRC=6.0, w_HR=1.0
    Learning rate η=1e-4, Batch size=32
Output: Trained model parameters θ

Initialize encoder and decoder parameters θ randomly
for epoch = 1 to N_epochs do
   for each batch (V, E, M, π*) in D do
      // MLE Forward Pass (Teacher Forcing)
      H ← TransformerEncoder(V, E, M; θ_enc)
      π̂_mle ← AttentionDecoder(H, M, π*; θ_dec)
      L_MLE ← -Σ_t log p_θ(π_t* | π*_<t, H)

      // RL Forward Pass (if λ > 0)
      π_greedy ← GreedyDecode(H, M; θ)
      π_sample ← SampleDecode(H, M; θ)
      R_greedy ← ComputeReward(π_greedy, π*)
      R_sample ← ComputeReward(π_sample, π*)
      R_combined ← (R_LSD^greedy - R_LSD^sample)
                 + w_KRC·(R_KRC^sample - R_KRC^greedy)
                 + w_HR·(R_HR^sample - R_HR^greedy)
      L_RL ← -R_combined·Σ_t log p_θ(π_t^sample | π_<t^sample, H)

      // Combined Loss and Update
      L_hybrid ← L_MLE + λ·L_RL
      ∇_θ L_hybrid ← BackPropagate(L_hybrid)
      θ ← θ - η·∇_θ L_hybrid
   end for

   // Validation
   Evaluate on validation set
   if validation_score improves then
      Save checkpoint
   end if
```

30: end for
31: return θ

4.5 Inference Procedure (Pseudo-code)

Algorithm 2 describes the inference procedure for route prediction:

Algorithm 2: Greedy Inference Procedure

Input: Node features V, Edge features E, Reachability mask M
Trained model parameters θ
Output: Predicted route sequence $\hat{\pi}$

// Encoding Phase
H ← TransformerEncoder(V, E, M; θ_enc)
$\bar{h}$ ← (1/N)·Σ_i h_i // Graph-level embedding

// Initialize Decoder
$\hat{\pi}$ ← [] // Empty route
M_t ← M // Current reachability mask
h_0, c_0 ← InitializeLSTM($\bar{h}$)

// Decoding Phase (Autoregressive)
for t = 1 to N do
    if M_t is all zeros then break // All nodes visited

    // LSTM Update
    x_t-1 ← Embed($\hat{\pi}$_t-1) if t > 1 else $\bar{h}$
    h_t, c_t ← LSTM(x_t-1, (h_t-1, c_t-1))

    // Multi-Glimpse Attention (K=2)
    g_t,0 ← h_t
    for k = 1 to K do
        u_i^(k) ← v^T·tanh(W_q^(k)·g_t,k-1 + W_r^(k)·h_i)
        α^(k) ← softmax(u^(k))
        g_t,k ← Σ_i α_i^(k)·h_i
    end for

    // Pointer Mechanism
    u_i^ptr ← v^T·tanh(W_q^ptr·g_t,K + W_r^ptr·h_i)
    u_i^ptr ← -∞ if M_t[i] = 0 // Mask infeasible nodes
    p_t ← softmax(u^ptr)

    // Select Next Node (Greedy)
    $\hat{\pi}$_t ← argmax_i p_t[i]
    Append $\hat{\pi}$_t to $\hat{\pi}$

    // Update Mask
    M_t[$\hat{\pi}$_t] ← 0 // Mark as visited

```
    if π̂_t is pickup node then
        M_t[corresponding_delivery] ← 1  // Enable delivery
    end if
end for

return π̂
```

5. Dataset

Experiments are carried out using the LaDe-P benchmark [9], which is a large-scale, publicly available last-mile delivery dataset built from actual courier operations in different Chinese cities. Three city partitions are employed, each displaying a distinct characteristic, irrespective of weather conditions:

Table 2: The framework was assessed using the LaDe dataset in the package pickup (LaDe-P) scenario:

City	Packages	Couriers	Characteristics
Shanghai	1.45 million	4,502	High-density, flat urban
Chongqing	1.17 million	2,982	Complex roads
Yantai	~1.0 million	~3,000	Medium-density
Total	10.67 million	21,000	6 months, 2021 data

The data is organized: 60% for training, 20% for validation, and 20% for testing. Routes are regulated such that no courier makes more than 25 stops. Each instance contains the following attributes: (1) node features, such as coordinates, time windows, and distance from the related courier; (2) edge features, such as routing distance and journey time; and (3) dynamic reachability masks, which are updated as tasks are completed.(4) Routes are filtered to contain between 5 and 100 delivery stops ($N \leq 100$), consistent with the model's max_task_num constraint.

5.1 Baseline Methods

We compare against eight baseline methods, ranging from traditional optimization to deep learning: (1) Time-Greedy [20]: Always selects the most urgent package (lowest T_ToD); (2) Distance-Greedy [21]: Always selects the nearest package (lowest D_sd); (3) OR-Tools [Google]: Heuristic optimization for shortest total distance; (4) LightGBM / OSquare [12]: XGBoost next-step prediction with 12 engineered features; (5) FDNET [22]: LSTM encoder + Pointer decoder for food delivery; (6) DeepRoute [7]: Transformer encoder (2021) with attention decoder; (7) Graph2Route [5]: Dynamic ST-GNN

5.2 Evaluation Metrics

- The performance using four metrics: Kendall Rank Correlation(KRC) [23], which measures global route order quality; Location Square Deviation (LSD) [24], which measures spatial accuracy; HR@k (Hit Rate at 3), which measures local next-stop accuracy [25]; and ED (Edit Distance) [26], which measures the minimum number of edits required to align predicted and actual routes.
- Kendall Rank Correlation (KRC): $KRC = (nc - nd) / (0.5 \cdot n \cdot (n-1))$, where nc and nd are concurring and conflicting pairs. Higher is better; 1.0 = perfect global ordering.
- Levenshtein Sequence Distance (LSD): Minimum edit operations (insertions, deletions, substitutions) to transform the predicted sequence into the ground truth. Lower is better; 0 = identical.
- Edit Distance (ED): Equivalent to LSD in this context. Lower is better.

- Hit Rate at 3 (HR@3): HR@3 = | Intersection (top-3 predicted, top-3 ground truth) | / 3. Measures local step-level prediction accuracy. Higher is better.

5.3 Baseline and Hyperparameters Implementation Details

The primary baseline is DRL4Route [9], the current state-of-the-art for last-mile route prediction on LaDe-P. HGTRL hyperparameters: d_model = 64, H = 8 attention heads, L = 2 Transformer layers, $\lambda = 0.3$ (RL weight), w_KRC = 6.0, w_HR=1.0 batch size = 32, learning rate = 1×10^{-3} with Adam optimiser, and beam width K = 5 at inference. Models are trained for up to 50 epochs, with an early halt (patience = 5) on validation KRC.

The model was implemented in PyTorch (compiled with CUDA 12.4) and trained on a multi-GPU environment utilizing three 8GB NVIDIA V100 GPUs. The training process employed the Adam optimizer with a learning rate of 0.001 and a dropout rate of 0.1. Due to the large scale of the datasets, the training procedure lasted approximately 24 hours per city.

Overall Performance Comparison

The GTRL framework was brutally benchmarked against a comprehensive array of 8 distinct baseline models, ranging from primitive algorithmic heuristics (Time-Greedy, Distance-Greedy), to commercial operations research solvers (Google OR-Tools), through to the previous apex of deep learning architectures (DeepRoute, Graph2Route, PAPN, and DRL4Route2).

The empirical results were nothing short of a paradigm shift. Across all three topographically diverse cities, the GTRL model established an absolute, undisputed dominant frontier:

- **Macroscopic Structural Fidelity (KRC Transformation):** Historically, deep learning models stagnated, battling to push KRC past the high 50% range (DRL4Route2 peaked at 57.23% in Chongqing). The GTRL architecture violently shattered this ceiling. The model achieved a jaw-dropping KRC of **71.55% in Chongqing, 72.70% in Shanghai, and 71.04% in Yantai**. This represents an absolute, linear improvement of **14.32% to 15.50%**. An improvement of 15 percentage points in a metric as mathematically rigid as Kendall Rank Correlation indicates a fundamental leap in the model's ability to comprehend long-range topographical structures.
- **Microscopic Spatial Deviation (LSD & ED Compression):** High KRC scores are meaningless if the actual physical coordinates of the route are scattered. Fortunately, the massive KRC leap corresponded directly with massive drops in spatial errors. The Location Square Deviation (LSD) compressed drastically by up to 16% (e.g., dropping from 3.06 to a highly accurate 2.56 in Shanghai). Correspondingly, absolute Edit Distance (ED) fell by -0.17 to -0.32 across the board. In a real-world dispatch center, this means human operators will have to manually correct significantly fewer algorithmic mistakes.
- **Localized Short-Term Predictive Robustness (HR@3):** A persistent fear in Reinforcement Learning optimization is that by forcing the model to focus heavily on the global macro-route (KRC), it loses its ability to accurately predict the immediate next step. The data absolutely refutes this fear. The localized prediction fidelity (HR@3) remained aggressively competitive, peaking at an exceptional 73.22% in Shanghai (a full percentage point higher than the previous best). This definitively proves that the hybrid MLE-RL objective function successfully optimized the global route without sacrificing short-term, micro-level accuracy.

Key Findings:

1. Global Ordering (KRC): Achieved 71.55% (Chongqing), 72.70% (Shanghai) and 71.04% (Yantai)-representing +14.32%, +15.50% and +15.10% improvements over DRL4Route2 (which scored 57.23%, 57.20%and 55.94% respectively). Significantly improving the global route structure by a gain of 14-15 points reflects a demonstrated capability to maintain superior route sequencing on a massive scale.

2. Spatial Deviation (LSD): Across the three cities, the LSD measurement decreased across the board: Chongqing (2.16 vs. 2.43, an 11% reduction), Shanghai (2.56 vs. 3.06, a 16% reduction) and Yantai (2.51 vs. 2.62, a 4% reduction). The LSD for Chongqing is the lowest overall value (2.16) due to the city's unique geography inherently limiting routing options.
3. Edit Distance (ED): Reductions of -0.22 (Chongqing), -0.32 (Shanghai) and -0.17 (Yantai) indicate that significantly less manual correction will be needed when predicting future routes, directly improving real-world operational efficiency.
4. Local Accuracy (HR@3): Rather than a strict transaction, HGTRL actually outperforms DRL4Route2 in Shanghai (73.22% vs. 72.18%, a +1.04% gain). In Chongqing and Yantai, the model experiences only marginal drops of -0.38% and -0.22%, respectively. This proves that the large global quality improvements (KRC, LSD, ED) were achieved without sacrificing local, step-by-step predictive accuracy.

Table 3: The comparative evaluation of HGTRL against DRL4Route across all three cities and four metrics. Note: ↑ = higher is better; ↓ = lower is better. Green cells = improvement; amber = marginal decline.

Method	**Chongqing (CQ)**				**Shanghai (SH)**				**Yantai (YT)**			
	HR@3 ↑	**KRC ↑**	**LSD ↓**	**ED ↓**	**HR@3 ↑**	**KRC ↑**	**LSD ↓**	**ED ↓**	**HR@3 ↑**	**KRC ↑**	**LSD ↓**	**ED ↓**
Time-Greedy	63.86	44.16	3.91	1.74	59.81	39.93	5.2	2.24	61.23	39.64	4.62	1.85
Distance-Greedy	62.99	41.48	4.22	1.6	61.07	42.84	5.35	1.94	62.34	40.82	4.49	1.64
Or-Tools	64.19	43.09	3.67	1.55	62.5	44.81	4.69	1.88	63.27	42.31	3.94	1.59
FDNET	69.98	52.07	3.36	1.51	69.05	52.72	4.08	1.86	69.08	50.62	3.6	1.57
DeepRoute	72.09	55.72	2.66	1.51	71.66	56.2	3.26	1.86	71.44	54.74	2.8	1.53
Graph2Route	72.31	56.08	2.53	1.5	71.69	56.53	3.12	1.86	71.52	55.02	2.71	1.54
PAPN	72.79	56.2	2.61	1.47	71.95	56.8	3.14	1.84	71.32	54.6	2.8	1.52
DRL4Route2	73.12	57.23	2.43	1.48	72.18	57.2	3.06	1.84	72.07	55.94	2.62	1.51
HGTRL	**72.74**	**71.55**	**2.16**	**1.26**	**73.22**	**72.7**	**2.56**	**1.52**	**71.85**	**71.04**	**2.51**	**1.34**
Improvement	**-0.38%**	**14.32%**	**-0.27**	**-0.2**	**1.04%**	**15.50%**	**-0.5**	**-0.3**	**-0.22%**	**15.10%**	**-0.11**	**-0.2**

5.3 Ablation: MLE vs. RL vs. Hybrid Training

Ablation Study: Deconstructing the Architectural Engine

To isolate the exact mechanical origins of these massive performance leaps, a highly targeted architectural deconstruction (ablation study) was executed. By surgically severing distinct components of the hybrid objective function, we definitively proved structural dependency.

1. **Severing Reinforcement Learning ($\lambda = 0$):** When the REINFORCE algorithm was deactivated, forcing the model to rely solely on supervised imitation learning (Teacher Forcing), the architectural performance immediately degraded. The KRC metric suffered a catastrophic collapse, dropping by a full 3.2 percentage points instantaneously. This empirical drop definitively proves that pure imitation learning suffers from compounding exposure bias, and that the RL component is an absolute structural necessity for pushing global accuracy boundaries past the 70% threshold.

2. **Severing Imitation Learning ($\lambda = 1$):** Conversely, attempting to train the massive Transformer architecture entirely via pure Reinforcement Learning—without the stabilizing prior of MLE—resulted in catastrophic gradient explosion. Operating in an action space of $100!$, the unguided RL agent could not locate the sparse reward signal, leading to severe training instability and effectively zero functional convergence. This proves that the hybrid approach is not merely an optimization trick; it is a fundamental prerequisite for training stability.

Codebase Implementation: This rigorous scientific isolation is easily replicable within the software stack by executing the train.py wrapper utilizing custom CLI arguments. By passing the flag --ablation True, the framework overrides configuration parameters, systematically disabling use_encoder and modulating the rl_ratio to perfectly mimic the isolated theoretical conditions described above, allowing seamless verification of the metric degradation.

To isolate each training component's contribution, HGTRL (λ=0.3) is compared against HGTRL-MLE (λ=0; teacher forcing only) and HGTRL-RL (λ=1; RL only). Across all cities, HGTRL-RL achieves higher KRC than HGTRL-MLE (approximately +8 pp) but exhibits substantially higher training instability and slower convergence. HGTRL (hybrid) achieves the best KRC and LSD simultaneously, confirming that imitation learning provides an essential initialisation prior that stabilises RL policy optimisation, while RL subsequently drives global sequence quality beyond the ceiling imposed by step-level imitation alone.

Table 4: Ablation and Fine-tuning Study Results

Method	**Chongqing (CQ)**				**Shanghai (SH)**				**Yantai (YT)**			
	HR@3 ↑	**KRC ↑**	**LSD ↓**	**ED ↓**	**HR@3 ↑**	**KRC ↑**	**LSD ↓**	**ED ↓**	**HR@3 ↑**	**KRC ↑**	**LSD ↓**	**ED ↓**
PAPN	72.79	56.2	2.61	1.47	71.95	56.8	3.14	1.84	71.32	54.6	2.8	1.52
DRL4Route2	73.12	57.23	2.43	1.48	72.18	57.2	3.06	1.84	72.07	55.94	2.62	1.51
HGTRL	**72.74**	**71.55**	**2.16**	**1.26**	**73.22**	**72.7**	**2.56**	**1.52**	**71.85**	**71.04**	**2.51**	**1.34**
Improvement	**-0.38%**	**14.32%**	**-0.27**	**-0.2**	**1.04%**	**15.50%**	**-0.5**	**-0.3**	**-0.22%**	**15.10%**	**-0.11**	**-0.2**

6. Discussion

6.1 Theoretical Implications

The principal theoretical finding is that the KRC-LSD reward decomposition provides a uniquely effective training signal: KRC penalises global ordering inversions while LSD penalises large positional displacements. Together, they incentivise a policy that is simultaneously globally coherent and locally conservative. The 15-point improvement in KRC over DRL4Route, achieved without sacrificing HR@3, demonstrates empirically that a Transformer encoder's capacity for global self-attention is the critical architectural ingredient for producing orderly full-day routes a capacity that GCN-based encoders, which aggregate only within local graph neighbourhoods, cannot replicate at comparable model scale.

6.2 Practical Implications

The KRC enhancement from 0.57 to 0.72 yields a major decrease in sequence inversions from an operational point of view. The baseline model (DRL4Route) misaligns about 21.5% of delivery pairings for a typical 50-stop route, while HGTRL minimizes this to 14%- 35% relative reduction in sequencing mistakes. Reducing out-of-order deliveries directly reduces re-delivery frequency and customer discontent because courier compensation is based on completion rates. Furthermore, the reduction in Edit Distance (from 1.52 to 1.26 in Chongqing, for example) suggests that human dispatchers need to make fewer manual corrections to created routes, which lowers operational overhead."

6.3 Limitations

Several limitations constrain the current study and motivate future work.

First, and most critically, the model does not incorporate real-time weather data and adverse weather conditions significantly alter courier routing behaviour by reducing accessible stops, slowing travel, increasing parcel damage risk, and forcing couriers to adopt modified routing patterns that deviate substantially from fair-weather training distributions. A weather-aware routing model that integrates meteorological forecasts as dynamic node or edge features represents a natural and practically important extension.

Second, the maximum node count of $N = 100$ is a hard architectural constraint that precludes application to couriers with larger daily task volumes, increasingly common as parcel density grows in megacities.

Third, the model is trained and evaluated on a single delivery company's operational data; generalisability to other logistics providers with different operational protocols has not been assessed.

Fourth, beam-search decoding introduces inference latency proportional to beam width, which may constrain real-time applications.

Fifth, the LaDe-P dataset does not capture all relevant courier decision factors, including informal task prioritisation rules and personal rest break preferences.

6.4 Future Research Directions

Future research should pursue the following directions. First, integrating weather, traffic, and time-of-day features as dynamic edge attributes would enable context-sensitive routing that adapts to real-world variability.

Second, scaling to $N > 100$ nodes via hierarchical clustering pre-processing or sparse attention mechanisms would extend applicability to high-volume routes.

Third, multi-task learning across city datasets could yield a universal routing model that generalises across urban geographies without city-specific fine-tuning.

Fourth, incorporating uncertainty quantification via Bayesian or collaborative approaches would enable the model to signal low-confidence predictions for dispatcher review.

Fifth, the framework could be extended to multi-courier (fleet-level) routing by incorporating vehicle assignment as an additional output head.

7. Conclusion

This paper presented the Hybrid Graph-Transformer Reinforcement Learning (HGTRL) model, a novel architecture for courier route prediction in last-mile urban delivery. By combining a multi-head Transformer encoder with an attention-based LSTM decoder and training under a composite MLE-RL objective that jointly optimises Kendall Rank Correlation and Levenshtein Sequence Distance, HGTRL achieves substantial and consistent improvements over the DRL4Route state-of-the-art across three geographically distinct Chinese cities.

The key empirical finding a 15 points improvement in global route ordering (KRC) with simultaneous reductions in sequence displacement (LSD) and edit distance (ED) demonstrates that the Transformer encoder's global self-attention, combined with a sequence-level reward signal, resolves the tension between step-level imitation accuracy and full-route ordering fidelity. The city-specific analysis reveals that urban density. Each differentially modulate evaluation metric trade-offs, providing practitioners with actionable deployment guidelines. As global parcel volumes continue to grow, data-driven route prediction models of the kind proposed here will play an increasingly central role in the intelligent logistics systems of the future.

REFERENCES

1. Joerss, M., F. Neuhaus, and J. Schröder, *How customer demands are reshaping last-mile delivery.* The McKinsey Quarterly, 2016. **17**: p. 1-5.
2. Kirkpatrick, S., C.D. Gelatt Jr, and M.P. Vecchi, *Optimization by simulated annealing.* science, 1983. **220**(4598): p. 671-680.
3. Dantzig, G.B. and J.H. Ramser, *The truck dispatching problem.* Management science, 1959. **6**(1): p. 80-91.
4. Cattaruzza, D., N. Absi, D. Feillet, and J. González-Feliu, *Vehicle routing problems for city logistics.* EURO Journal on Transportation and Logistics, 2017. **6**(1): p. 51-79.
5. Wen, H., et al. *Graph2route: A dynamic spatial-temporal graph neural network for pick-up and delivery route prediction*. in *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 2022.
6. Mao, X., et al. *Drl4route: A deep reinforcement learning framework for pick-up and delivery route prediction*. in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2023.
7. Wen, H., et al. *Package pick-up route prediction via modeling couriers' spatial-temporal behaviors*. in *2021 IEEE 37th International conference on data engineering (ICDE)*. 2021. IEEE.
8. Zhang, Y., et al., *Route prediction for instant delivery.* Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2019. **3**(3): p. 1-25.
9. Wu, L., et al., *LaDe: The first comprehensive last-mile delivery dataset from industry.* arXiv preprint arXiv:2306.10675, 2023.
10. Toth, P. and D. Vigo, *Vehicle routing: problems, methods, and applications*. 2014: SIAM.
11. Gendreau, M., G. Laporte, and D. Vigo, *Heuristics for the traveling salesman problem with pickup and delivery.* Computers & operations research, 1999. **26**(7): p. 699-714.
12. LightGBM, G.K., *A Highly Efficient Gradient Boosting Decision Tree.* Garnett IGaUVLaSBaHWaRFaSVaR, editor, 2017.
13. Chandrashekar, G. and F. Sahin, *A survey on feature selection methods.* Computers & electrical engineering, 2014. **40**(1): p. 16-28.
14. Kipf, T.N. and M. Welling, *Semi-supervised classification with graph convolutional networks.* arXiv preprint arXiv:1609.02907, 2016.
15. Wu, Z., et al., *A comprehensive survey on graph neural networks.* IEEE transactions on neural networks and learning systems, 2020. **32**(1): p. 4-24.
16. Vaswani, A., et al., *Attention is all you need.* Advances in neural information processing systems, 2017. **30**.
17. Hochreiter, S. and J. Schmidhuber, *Long short-term memory.* Neural computation, 1997. **9**(8): p. 1735-1780.
18. Vinyals, O., M. Fortunato, and N. Jaitly, *Pointer networks.* Advances in neural information processing systems, 2015. **28**.
19. Bello, I., et al., *Neural combinatorial optimization with reinforcement learning.* arXiv preprint arXiv:1611.09940, 2016.
20. Boysen, N., S. Fedtke, and S. Schwerdfeger, *Last-mile delivery concepts: a survey from an operational research perspective.* Or Spectrum, 2020. **43**(1): p. 1-58.
21. Li, Y. and T.-P. Tan, *DiffRoute: An End-to-End Framework for Multi-Step Delivery Route Prediction.* IEEE Access, 2026.
22. Savelsbergh, M.W. and M. Sol, *The general pickup and delivery problem.* Transportation science, 1995. **29**(1): p. 17-29.
23. Abdi, H., *The Kendall rank correlation coefficient.* Encyclopedia of measurement and statistics, 2007. **2**: p. 508-510.
24. Sievers, G.L., *Estimates of location: a large deviation comparison.* The Annals of Statistics, 1978: p. 610-618.
25. Li, D., R. Jin, J. Gao, and Z. Liu. *On sampling top-k recommendation evaluation*. in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020.
26. Ristad, E.S. and P.N. Yianilos, *Learning string-edit distance.* IEEE Transactions on Pattern analysis and machine intelligence, 2002. **20**(5): p. 522-532.