Double Deep Q-Network with Experience Replay for Time Dependent Vehicle Routing Problem with Time Windows Under Historical Congestion Constraints

Rina Refianti, Alifurrohman*, Eri Prasetyo Wibowo, Ina Siti Hasanah, Achmad Benny Mutiara
Gunadarma University, Depok, Indonesia
E-mail: alifurrohman@staff.gunadarma.ac.id
*Corresponding author

Keywords: Deep Q-Network (DQN), time dependent vehicle routing problem with time windows (TD-VRPTW), historical congestion, route optimization, logistics distribution, Jakarta

Received: September 26, 2025

This study addresses the Time-Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) for a single-vehicle urban distribution system in Jakarta. Time-dependent travel times are constructed from one week of hourly historical congestion profiles obtained from TomTom Traffic and preprocessed into time-varying speed factors that are mapped to 40 and 50 customer delivery instances with a common service window of 08:00–19:00. A Deep Q-Network (DQN) enhanced with Double DQN and Prioritized Experience Replay (PER) is trained end-to-end using a multilayer perceptron with two hidden layers (128 and 64 units, ReLU activations) to approximate the state–action value function. The reward function penalizes time-dependent travel time, lateness with respect to customer time windows, long inter-customer jumps, and inter-cluster moves, thereby shaping the policy toward both schedule adherence and congestion-aware routing. For each scenario, the agent is trained for 1,000 episodes under three random seeds and evaluated on three representative weekdays (Monday, Wednesday, and Friday). Across all settings, the learned policy achieves a 100% on-time delivery rate with zero late customers, with best time-dependent route costs of approximately 526–539 minutes for 40 customers and 595–617 minutes for 50 customers. Comparative experiments with Genetic Algorithm (GA) and Ant Colony Optimization (ACO) show that ACO attains the shortest travel times, while the proposed DQN+PER model yields routes that are only about 5–8% longer than ACO but reduce time-dependent travel cost by roughly 35–45% compared with GA in the same TD-VRPTW instances. Reward and loss trajectories exhibit smooth convergence, and a sensitivity analysis on the lateness penalty confirms that the main conclusions are robust to hyperparameter variations. These findings demonstrate that leveraging historical congestion to build time-dependent travel times enables DQN-based control to produce competitive, congestion-aware solutions for TD-VRPTW in realistic urban distribution networks.

Povzetek: Ta študija predlaga model DQN, opremljen z Double DQN in Prioritized Experience Replay (PER), za reševanje časovno odvisnega problema usmerjanja vozil s časovnimi okni (TD-VRPTW) v mestni distribuciji na podlagi zgodovinskih podatkov o prometnih zastojih.

1 Introduction

Digital transformation in the urban logistics sector has raised expectations for distribution speed and accuracy, especially in areas with dynamic demand and high congestion levels such as Jakarta. Parcel delivery firms, couriers, and freight logistics now face the complex challenge of designing vehicle routes that are efficient while meeting service time constraints (time windows). Densely populated urban areas, road networks that are not always optimally structured, and daily fluctuations in traffic volume create distinct challenges for efficient route planning. Failure to meet these constraints not only affects customer satisfaction, but also incurs additional costs in the form of fuel consumption, late-delivery penalties, and increased carbon emissions [1] [2] [3]. Several recent industry reports and academic surveys underline that route decisions that are not adaptive to actual traffic conditions and demand can cause logistics inefficiencies of up to 20–30% from optimal potential [4]. Consequently, time-aware (time-dependent) vehicle-routing models are essential to close the gap between planned and realized performance.

In general, the Vehicle Routing Problem (VRP) is a classical combinatorial optimization problem that aims to determine delivery routes for a set of vehicles serving customers from one or more depots. Over time, VRP has evolved into more realistic variants such as the Time-Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW), which explicitly considers variations in travel time due to traffic congestion and imposes time-window constraints for customer service. In TD-VRPTW, travel times between locations are not fixed but depend on the departure time, reflecting fluctuating traffic

conditions, a factor especially relevant in urban logistics or dense road networks [5] [6]. Many heuristics and metaheuristics assume static conditions and fail to account for dynamic changes in traffic or customer demand, which can lead to suboptimal routing decisions due to entrapment in local optima [7] [8].

In this context, Reinforcement Learning (RL) emerges as a promising paradigm to overcome these limitations. Unlike traditional methods, RL enables a learning agent to explore and exploit the environment through experience, thereby producing adaptive policies without requiring explicit rules [9]. One of the most relevant RL approaches for large-scale dynamic routing is the Deep Q-Network (DQN) [10]. The DQN algorithm uses neural networks to estimate Q-values, making vehicle-routing decision making more efficient and stable than classical Q-learning [11]. Across domains such as Atari games [12], robotics [13], and route optimization [14] DQN has been shown to be a stable and efficient reinforcement-learning baseline [15]. Early applications of DQN to VRP, although still limited, have demonstrated competitive performance and mark the beginning of serious exploration of this algorithm in urban logistics [10].

However, most studies are still limited to idealized simulation environments. Few have integrated actual or historical traffic congestion data as an explicit component in DQN decision-making. Therefore, this research aims to investigate and develop a DQN model that integrates traffic congestion data as a key factor in the objective function to optimize logistics distribution routes in Jakarta. The proposed model is expected to enhance distribution efficiency by adaptively responding to urban traffic dynamics.

2 Related work

Research on the Vehicle Routing Problem (VRP) and its derivatives particularly the Time-Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) has long been a major topic in the field of transportation and logistics. Classical heuristic and metaheuristic methods such as Genetic Algorithm (GA), Tabu Search, and Ant Colony Optimization (ACO) are widely used because of their ability to produce near-optimal solutions with relatively short computation times. Recent TD-VRPTW variants have even incorporated road network realism and environmental costs: Liu et al. [16] proposed a TD-VRPTW model with a congestion avoidance approach using Improved ACO (IACACAA), which is able to reduce total distribution costs, fuel consumption, and CO_2 emissions by adjusting departure times to avoid peak hours. Fan et al. [17] introduced Hybrid GA + VNS (HGAVNS-TS), which takes into account fuel and emission consumption through the MEET model for multi-depot scenarios with time-dependent speeds, showing small optimality gaps and fast convergence. Meanwhile, Wang et al. [18] examined TDEVRPTW with path flexibility using Variable Neighborhood Search (VNS) to describe the importance of route selection that is sensitive to time and changing traffic conditions.

On the side of exact methods and road network based approaches, Ben Ticha et al. [5] developed a Branch and Price approach directly on time-dependent road networks, which proved to produce more accurate solutions compared to customer graph based representations and to avoid false infeasibility cases. Gmira et al. [6] applied Tabu Search with a dominant shortest path structure that is able to evaluate travel time efficiently on complex road networks, achieving an optimality gap below 1% with computation times that are much faster than exact methods. For large-scale cases, Cruz-Chávez et al. [19] developed a parallel Grid-Based Genetic Algorithm (GGA) that showed improvements approaching the best solutions on the Gehring and Homberger benchmarks with significant speedup. However, these conventional approaches still have limitations in terms of adaptability to rapidly changing urban traffic dynamics and require manual parameter adjustments whenever scenarios change [20].

The advancement of Reinforcement Learning (RL) offers a more flexible alternative. RL enables a learning agent to explore and exploit the environment to determine an optimal routing policy without requiring explicit rules. The application of Deep Q-Network (DQN), which combines Q-learning with deep neural networks, has proven effective across various domains, including logistics route planning [21] [11]. DQN's advantage lies in its ability to learn effective sequences of environmental transitions, allowing the exploration of large solution spaces and improving cost efficiency in logistics scenarios [2]. Yue et al. [22] extended this concept by proposing a Deep Q-Learning-Based Multi-Objective Evolutionary Algorithm (DQMOEA) for TD-GVRPTW, which combines DQN-based adaptive local search within an evolutionary framework. The results show significant performance improvements on the Hypervolume (HV) and Inverted Generational Distance (IGD) metrics compared to four other strong multi-objective algorithms.

Further research on the use of Double DQN and Prioritized Experience Replay (PER) have proven effective in improving training stability and efficiency across various route-optimization scenarios. The development of Double Q-Learning with PER for traffic-route simulation significantly reduces overestimation bias and speeds up convergence [23]. Zhu et al. [24] employed a Dueling Double DQN with PER for unmanned vehicle path planning, improving training stability and accuracy. Huo [25] applied DQN-PER to multi-objective routing, accelerating convergence and avoiding local optima. Niu et al. [26] showed that DDQN-PER increases navigation efficiency for ships in multi-agent settings.

The application of DQN in vehicle route planning also shows high effectiveness in dealing with congestion. The model can adapt routing decisions based on real-time as well as historical traffic conditions. For example, DQN has been used to optimize autonomous vehicle routes in heavy traffic [27], reduce queues by up to 49% in urban environments [25], and lower delay in QoS-based multipath routing [28]. These results affirm that integrating DQN with congestion information can

significantly improve the efficiency of intelligent transportation.

The application of Deep Q-Networks (DQN) under Time Window constraints has shown high effectiveness in producing adaptive and efficient solutions. Hybrid models such as Greedy-DQN have been proven to significantly reduce the number of vehicles and total travel distance [11], while attention-based encoder decoder approaches accelerate large-scale optimization [10] . These studies indicate that DQN is effective for solving complex routing problems.

Table 1: Comparison of related research

Name	Method	Dataset Used	Key Metrics	Result
Wang et al. (2020) — TDEVRPTW-PF	Improved Variable Neighborhood Search (VNS) with Path Flexibility and Virtual Distribution Center Concept	30 customers, 12 charging stations, and 5 EVs; working hours from 06:00 to 22:00; speeds of 50–60 km/h (varying by time and road segment).	Minimize the total travel distance.	The VNS (maximum 300 iterations) achieved a best total travel distance of 1,512.4 km, with 30 customers served by 5 EVs.
Jie et al. (2022) — TDVRP (soft TW + stochastic)	Hybrid: Sweep + Improved PSO (IPSO), incorporating traffic congestion and weather conditions.	Solomon C101/C102/R101/R102/RC101/RC102 benchmark instances with time-dependent speeds (Donati, 2008); cost and penalty coefficients: 150 per day, 2 per kilometer, 0.5 for early arrivals, and 1.5 for late arrivals.	distance, waiting time, runtime.	Minimal K; distance deviation of ~8.85% vs. the benchmark and better than NNC/FHI/ACO; waiting time is significantly lower; CPU time ≈ 395–406 s.
Fan et al. (2021) — TDMDGVRPTW	HGAVNS_TS (Hybrid GA + VNS) with temporal–spatial distance clustering, SA-based acceptance, and adaptive neighborhoods.	Benchmark MDVRP and MDVRPTW; road speeds are time-dependent (modeled using trigonometric functions); fuel consumption is affected by speed, load, and road gradient.	Objective: Total Cost = fuel + fixed costs + early/late penalties; evaluation based on the optimality gap vs. ALNS/HGSADC/VNS and runtime.	Effective: achieves small optimality gaps and faster convergence than baseline methods on MDVRP/MDVRPTW instances.
Gmira et al., 2020 (EJOR)	Tabu Search with dominant shortest-path strategy, constant-time neighborhood evaluation, tabu list, aspiration criteria, and diversification mechanisms.	NEWLET instances (narrow and wide time windows) for road-network TD-VRPTW, with up to 200 nodes and 580 arcs.	Impr (%) vs greedy; GapO (%) vs optimum; Time(s) TS vs BP	Improvement of 3.04% (narrow) and 5.28% (wide); optimality gap GapO < 1%; average TS runtime of 5.43 s vs. 358.27 s (narrow) and 13.14 s vs. 2454.46 s (wide) compared with BP.

Name	Method	Dataset Used	Key Metrics	Result
Cruz-Chávez et al. (2019) — Grid-Based GA for VRPTW	GGA (grid-based genetic algorithm) built on GA-VRPTW: global selection and k-point crossover, with segment-wise mutations distributed via MPI on a MiniGrid architecture.	Solomon C101–C109 (100 cust.) and Gehring–Homberger C1_10_x (1000 cust.).	RE (relative error) vs optimal/BKS, runtime, speedup, latency.	GGA consistently attains solutions closer to the optimum than GA-VRPTW (Wilcoxon p = 0.003843); runtimes are on the order of several hundred seconds, with speedups approaching or even exceeding linear in several configurations.
Liu et al. (2019) — TD-VRPTW + Congestion Avoidance	IACACAA (Improved Ant Colony Algorithm) combined with OFVTPBN, which optimizes the inter-node travel process by scheduling departure times and allowing vehicles to stop temporarily during congestion.	Solomon C1/R2/RC1/RC2 benchmark instances with 100 customers; depot operating from 07:00 to 23:00; 10-minute time slots; peak-hour congestion from 08:00–09:00 and 18:00–19:00; time-dependent speeds of 20 km/h under congestion and 60 km/h (with variations) under normal conditions.	total distribution cost, driver cost, fuel + emission cost, CO_2 (kg), travel time (min), distance (km), number of vehicles, runtime (s).	Compared with classical ACO (TACA), total cost is reduced by up to ~43% (e.g., R204/R205/R206/R210); fuel/emission costs and CO_2 decrease by ~60%; runtime is ≤ ~403 s; for instance, in RC208 a feasible route with 6 vehicles is obtained, with travel time shorter relative to the travelled distance, indicating effective congestion avoidance.
Yue et al. (2024) — TD-GVRPTW (multi-obj.)	DQMOEA + DQN-based adaptive local search	Solomon-derived instances (25/50/100 customers); 5 time zones; soft time windows.	Hypervolume, Inverted generational distance	Outperforms NSGA-III, MOEA-D, MaOEA-AC, and hpaEA, achieving dominance on 61.3% of the instances in terms of HV and 64.8% in terms of IGD (out of 168 instances).

3 Methodology

3.1 Problem formulation

The problem considered in this study is formulated as a Time-Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) for a single vehicle. The objective is to minimize the total routing cost composed of (i) travel time that depends on the realized departure time, (ii) waiting time (early arrival before the opening of the time window), and (iii) lateness penalties (service start

after the closing time). The objective and constraints are given in Eqs. (1)–(7).

We consider a directed graph $G = (V, A)$with location set $V = (0) \cup V_c$, where node 0is the depot and $V_c = ($are customers; arcs $(i, j) \in A$satisfy $i \neq j$. Each customer $i \in V_c$has a service time window $[a_i, b_i]$and a service duration s_i(minutes). The time-dependent travel time from $i$to $j$when departing at time $t$is denoted $\tau_{ij}(t)$; in all constraints and the objective, $\tau_{ij}(\cdot)$is evaluated at the realized departure time. Routing decisions are encoded by binary variables $x_{ij} \in (0,1)$indicating whether arc $(i, j)$is traversed, along with the arrival time $t_i \geq 0$, waiting time $w_i \geq 0$(if arriving before a_i), and lateness $L_i \geq 0$(if service starts after b_i). We define the service start at node $i$as $t_i^{\text{start}} = t_i + w_i$and the departure time as $d_i = t_i + w_i + s_i$. Waiting and lateness enter the cost with non-negative weights $\alpha_w$and α_L(minute equivalents).

The optimization objective is to minimize the total routing cost, defined as the sum of time-dependent travel time, waiting time, and lateness, each evaluated at the realized departure times.

$$\min cost = \underbrace{\sum_{(i,j)\in A} x_{ij}\tau_{ij}(d_i)}_{time-dependent\ ravel\ time} + \alpha\omega \underbrace{\sum_{i\epsilon V_c} \omega_i}_{waiting} + \alpha L \underbrace{\sum_{i\epsilon V_c} L_i}_{latenes}, \; d_i = t_i + \omega_i + s_i (1)$$

The model is subject to several groups of constraints. First, the visit constraint ensures that each customer is served exactly once:

$$\sum_{j\in V(i)} x_{ij} = 1, \forall i \in V_c, \; \sum_{i\in V(j)} x_{ij} = 1, \forall j \in V_c \; (2)$$

Because we consider a single-vehicle tour, the route starts and ends at the depot.

$$\sum_{j\in V_c} x_{0j} = 1, \sum_{i\in v_c} x_{i0} = 1. \; (3)$$

Temporal consistency is captured by a time-propagation constraint that advances the next arrival by the realized departure plus the time-dependent travel time evaluated at that departure; a sufficiently large constant M(*big-M*) relaxes this constraint when the arc is not selected

$$t_j \geq d_i + \tau_{ij}(d_i) - M\big(1 - x_{ij}\big), \forall (i,j) \in A, i \neq j, \; (4)$$

Time-window compliance requires that service start within the window up to tolerated lateness, with non-negativity for waiting and lateness:

$$t_i^{start} = t_i + \omega_i \geq a_{i,} \; t_i^{start} \leq b_i + L_i, \forall \in V_c, (5)$$

$$\omega_i \geq 0, L_i \geq 0, \forall_i \in V_c \; (6)$$

Finally, the variable domains are stated explicitly.

$$x_{ij} \in \{0,1\}, t_i \geq 0 \, , \forall (i,j) \in A, i \neq j; \; t_i \geq 0 \; \forall i \in V; \; \omega_i \geq 0, L_i \geq 0 \; \forall i \in V_c \; (7)$$

To align the learning signal with the optimization goal in (1), the per-step reward used in training is the negative of the same cost components, each evaluated at the realized departure time $d_{i_t} = t_{i_t} + w_{i_t} + s_{i_t}$.

$$r_t = -\big(\tau i_t j_t\big(d_{i_t}\big) + \alpha_\omega \omega_{j_t} + \alpha_L L_{j_t}\big) - \lambda_{cc} 1\{k(i_t) \neq k(j_t), i_t \neq 0, j_t \neq 0\} \; (8)$$

In our experiments, the penalty weights and reward normalization are fixed. Lateness is weighted by $\omega_L = 400$ to strongly discourage time-window violations, while long jumps between customer groups are penalized through a jump penalty $\alpha_{\text{jump}} = 50$ seconds per kilometer and a cross-cluster penalty $\gamma_{\text{cross}} = 300$ seconds. Within the same cluster, short intra-cluster moves receive a smaller distance penalty $\beta_{\text{inner}} = 30$. The per-step reward is divided by a constant scale $S = 1000$ to keep reward magnitudes in a moderate range and maintain numerical stability during training. These coefficients were chosen based on preliminary sensitivity checks and then kept fixed across all scenarios rather than being extensively tuned as hyperparameters.

3.2 Data collection

In this study, the data collected include:

1. Customer Location Data

   Comprising two scenarios with 40 and 50 customer locations across Jakarta. The dataset contains the full addresses of each customer.

2. Traffic Congestion Data

   Information on congestion-prone locations in Jakarta obtained from news reports and publicly available data on official government website.

3.3 Data preprocessing

Customer addresses and congestion-prone points were first converted into geographic coordinates (latitude and longitude) using the TomTom Geocoding API. We used the TomTom Traffic API to obtain current and free-flow speeds per road segment together with day-of-week and hour-of-day metadata. From a one-week log we constructed hourly congestion factors at the rush-hour windows (e.g., 07:00–10:00 and 15:00–21:00) by taking a robust ratio of free-flow to current speed per segment, while non-rush hours are treated as neutral (factor ≈ 1). Each customer node inherits the hourly factor of its nearest road segment within a small radius. When computing travel for a leg $(i \rightarrow j)$, we evaluate the factor at the realized departure hour and, if the leg crosses an hour boundary, we accumulate the contribution hour by hour. These hourly rush-focused profiles define the time-dependent travel times used throughout the study.

3.4 Model DQN

The proposed model adopts a Deep Q-Network (DQN) to solve the Time-Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) under congestion conditions in Jakarta. DQN is chosen for its ability to learn end-to-end routing policies via reinforcement learning without requiring an explicit traffic dynamics model. The time-dependent travel times $\tau_{ij}(t)$are constructed from

hourly historical congestion and are evaluated at the realized departure time $d_i = t_i + w_i + s_i$.

The Q-network is implemented as a fully connected multilayer perceptron (MLP) that operates on node-wise features. At each decision step, the state is represented as an $N \times 6$ tensor, where N denotes the number of nodes (one depot and all customers), and each node has six hand-crafted features (visit status, current-node indicator, normalized distance to depot and cluster centroid, normalized current time, and residual slack to the closing time window). The network consists of two hidden layers with 128 and 64 units, respectively, each followed by a ReLU activation. The output layer is linear and produces one scalar Q-value per node, corresponding to the estimated return of selecting that node as the next visit. No dropout or batch normalization is applied in this architecture. The network is trained using the Adam optimizer with a learning rate of 1×10^{-4}. Preliminary experiments indicated that this relatively small and shallow architecture, combined with Double DQN and prioritized replay, yields stable training without noticeable overfitting.

Agent environment interaction experiences are stored in a prioritized experience replay (PER) buffer. We employ the proportional variant of PER (Schaul et al., 2016), in which each transition $i$is assigned a non-negative priority $p_i = | \delta_i | + \varepsilon$, where $\delta_i$is its temporal-difference (TD) error and $\varepsilon = 10^{-3}$ avoids zero priority. The sampling probability is defined as

$$P(i) = \frac{p_i^{\alpha}}{\sum_j p_j^{\alpha}} \quad (9)$$

where the exponent $\alpha \in [0,1]$controls the strength of prioritization. We set $\alpha = 0.6$, which provides a compromise between uniform replay ($\alpha = 0$) and fully greedy replay ($\alpha = 1$), so that transitions with larger TD errors are sampled more frequently while still maintaining sufficient diversity in the replayed experiences. To correct for the bias introduced by non-uniform sampling, we apply importance-sampling weights $w_i \propto (N \cdot P(i))^{-\beta}$, where $N$is the current buffer size. The exponent $\beta$is annealed linearly from $\beta_{\text{start}} = 0.4$at the beginning of training to $\beta_{\text{end}} = 1.0$at the end, so that the updates become increasingly unbiased as learning progresses. During training, the agent selects actions using an ε-greedy strategy, where $\varepsilon$is decayed from 1.0 to 0.05 over episodes to gradually shift from exploration to exploitation.

The reward function is designed to balance route efficiency and punctuality and is aligned with the TD-VRPTW objective described in Section 3.1. At each decision step, the reward is defined as the negative of the time-dependent travel, waiting, and service times, augmented with penalty terms for service lateness, long jumps between distant customers, and inefficient cross-cluster moves, and normalized by a constant reward scale to keep the magnitude of rewards in a numerically stable range (see Eq. (8)). The associated penalty weights and normalization factor are fixed across all experiments, based on preliminary sensitivity checks, so that different scenarios remain comparable.

3.5 Training strategy and experiment setup

The model training process was carried out in Python using the TensorFlow library. The training strategy was designed to maximize the ability of the Deep Q-Network (DQN) to learn efficient routing policies. Each training scenario was run for 1,000 episodes, which based on preliminary tests was sufficient to achieve convergence without overfitting. The main hyperparameter values used include a learning rate of 1×10^{-4} for the Adam optimizer, a discount factor $\gamma = 0.95$, a mini-batch size of 64, a replay buffer capacity of 50,000 transitions, a target-network update frequency of every 1,000 training steps, and at least 1,000 warm-up steps before training starts. Exploration followed an ε-greedy strategy, starting from $\varepsilon = 1.0$and decreasing to 0.05 using a multiplicative decay factor of 0.997 to balance exploration and exploitation during training. These hyperparameters were selected by combining common practice in DQN-based control with small-scale preliminary experiments on the Jakarta dataset: higher learning rates (e.g., 10^{-3}) produced unstable Q-value updates, while smaller values slowed convergence; $\gamma = 0.95$provided a reasonable trade-off between short-term routing costs and long-term episode returns; and a mini-batch size of 64 offered a compromise between noisy gradients for very small batches and increased memory usage for larger batches. Once this configuration was found to yield stable learning and reasonable convergence speed, it was kept fixed across all experimental scenarios rather than being extensively tuned through large-scale hyperparameter searches.

To ensure result stability, each scenario combination was run with three different random seeds (42, 77, 99). In addition, tests were conducted at two customer scales 40 and 50 customers to evaluate the effect of distribution density on route performance.

3.6 Evaluation metrics

The success of the Deep Q-Network (DQN) on the Time Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) is evaluated using five key metrics: Total Route Cost (total route travel time), On-time Delivery Ratio measuring delivery punctuality within the 08:00–19:00 window, Average Lateness assessing mean delay per customer, Distribution of Congestion Status indicating the model's ability to avoid congestion, and Route Visualization on an OpenStreetMap (OSM) map to verify visit order and traffic conditions.

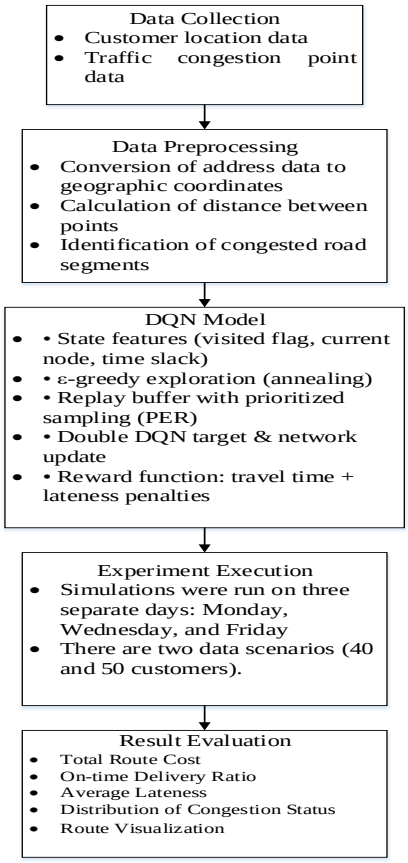

Figure 1: Research flowchart

4 Experiment and discussion

4.1 Dataset description and experimental scenarios

This study evaluates the performance of a Deep Q-Network (DQN) equipped with Double DQN and Prioritized Experience Replay (PER) on the Time Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) using two Jakarta delivery instances with 40 and 50 customers, respectively. Each instance consists of a single depot and multiple customer locations served within a common time window of 08:00–19:00 WIB. Historical congestion data were obtained from previous data collection using the TomTom Traffic API, which provides actual traffic speed and free-flow speed; the ratio of the two is used as an hourly congestion factor. Experiments are conducted for three weekdays: Monday, Wednesday, and Friday. In addition, each customer-size setting is tested with three random seeds (42, 77, 99) to ensure training stability and minimize initialization bias. The analysis focuses on best cost (minimum total travel time), on-time delivery percentage, average lateness, and the distribution of congestion status (successfully avoided, congested, and neutral).

4.2 Reward and loss analysis

This subsection presents an analysis of the reward and loss curves as an overview of the DQN learning process for the 40- and 50-customer cases. The graphs display the evolution of per-episode reward and average per-episode loss over 1,000 training episodes on three test days Monday, Wednesday, and Friday using the best seed, to assess the model's stability and convergence.

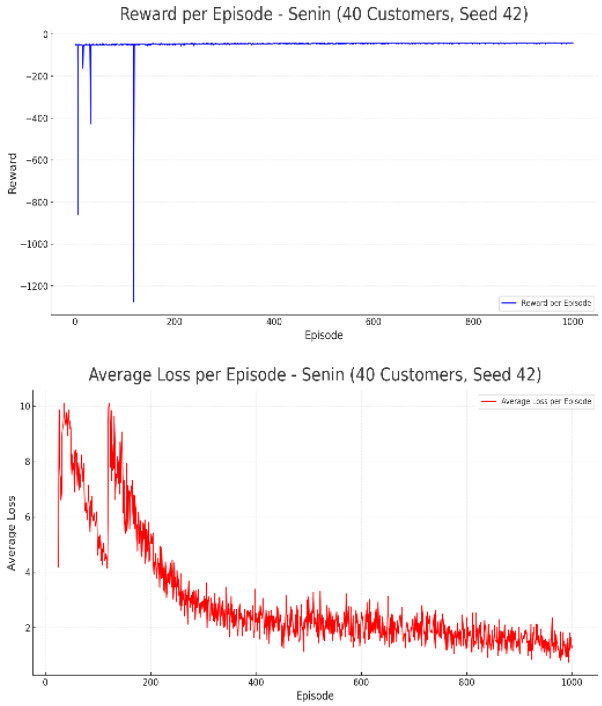

Figure 2: Training reward and loss curves for the 40-customer case (monday)

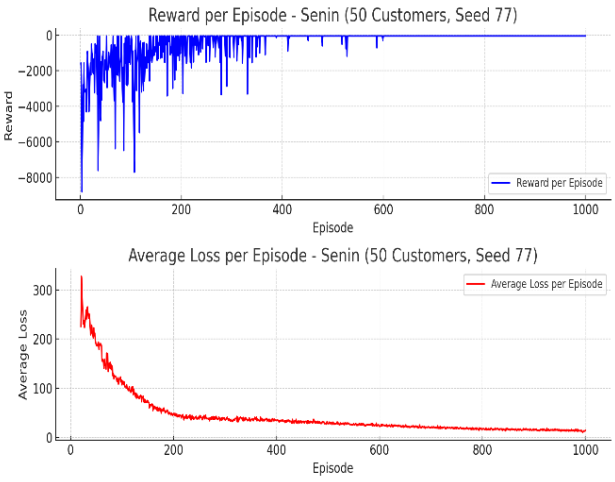

Figure 3: Training reward and loss curves for the 50 customer case (monday)

Figure 2. The per-episode reward and loss curves for the 40 customer case show a stable convergence pattern. At the beginning of training, the per-episode reward exhibits sharp negative fluctuations, indicating intensive exploration. After around 200 episodes, the per-episode reward rises and stabilizes within a narrow band around zero, indicating that the agent has reached a relatively stable policy. The loss pattern also shows a consistent downward trend: the initially high average loss gradually decreases and flattens out near episode 1,000. This combination of steadily improving reward and declining

loss indicates that the successfully learned an efficient routing policy that satisfies the time-window constraints for the 40 customer scenario.

Figure 3**.** In the 50 customer scenario, the training pattern poses greater challenges yet still trends toward good convergence. The per-episode reward is initially highly volatile and far below zero, with sharp drops approaching −8000, indicating increased complexity due to the larger number of delivery points. In the 50 customer case, the per-episode reward gradually improves after roughly 300 episodes and then stabilizes near zero, which suggests that the learning process converges despite the increased problem complexity. The per-episode loss falls steeply from a very high initial value (around 300) and stabilizes near zero in the later episodes. The consistent loss reduction together with the reward increase confirms the model's ability to discover an optimal policy, although the larger customer set requires a longer training process before stability is achieved.

4.3 Best route analysis and trip summary

This subsection presents the analysis of the best routes produced by the Deep Q-Network (DQN) model for each combination of day and customer count. The experimental results are consolidated in Table 2 to provide a comprehensive overview of the model's performance.

Table 2: Summary of results

Day	Customers	Seed	Best Cost (min)	On-time (%)	n Late	Congestion Hit	Congestion Avoided	Neutral Segments
Monday	40	42	529.4	100	0	10	5	26
Monday	40	77	529.4	100	0	5	10	26
Monday	40	99	529.4	100	0	5	10	26
Wednesday	40	42	526.6	100	0	5	7	29
Wednesday	40	77	526.6	100	0	5	7	29
Wednesday	40	99	526.6	100	0	5	7	29
Friday	40	42	538.7	100	0	2	10	29
Friday	40	77	538.7	100	0	2	10	29
Friday	40	99	538.7	100	0	2	10	29
Monday	50	42	617	100	0	8	9	34
Monday	50	77	616.9	100	0	10	8	33
Monday	50	99	606.9	100	0	9	9	33
Wednesday	50	42	611.4	100	0	8	7	36
Wednesday	50	77	595.5	100	0	5	9	37
Wednesday	50	99	602	100	0	9	6	36
Friday	50	42	616.4	100	0	7	8	36
Friday	50	77	614.5	100	0	2	12	37
Friday	50	99	619.3	100	0	7	8	36

Based on Table 2, in the 40 customer scenario the DQN model exhibits very consistent behaviour across the three random seeds for each day. The best route cost is identical for all seeds on a given day (529.4 minutes on Monday, 526.6 minutes on Wednesday, and 538.7 minutes on Friday), and every run achieves 100% on-time delivery with zero late customers. The congestion statistics also remain in a narrow range: on Monday each route encounters between 5 and 10 congested segments and successfully avoids 5–10 segments, on Wednesday 5 congested and 7 avoided segments are observed for all seeds, and on Friday the model typically hits only 2 congested segments while avoiding 10. These results indicate that the learned policy is highly stable in the 40 customer case, producing almost identical costs and schedule adherence while showing a consistent pattern in how congestion hotspots are managed.

In the 50 customer scenario, the DQN model again maintains 100% on-time delivery for all seeds and all days, confirming that the time-window constraints remain fully satisfied even as the problem size increases. The best route cost shows only modest variation between seeds, with values in the range of 606.9–617.0 minutes on Monday, 595.5–611.4 minutes on Wednesday, and 614.5–619.3 minutes on Friday. The number of congestion hits remains within approximately 2–10 segments per route, while the number of avoided segments is consistently high (6–12 segments), indicating that the policy is able to adapt its visiting order to more complex traffic conditions without sacrificing punctuality. Overall, the per-seed trip summaries confirm that the DQN model yields robust and reliable routes across different initialisations, days, and customer scales.

4.4 Sensitivity analysis

A complementary sensitivity analysis was also carried out on the lateness penalty parameter β_late in the reward function, rather than on the Prioritized Experience Replay (PER) hyperparameters. This test was performed on the most challenging scenario (50 customers on Friday) by varying $\beta\_late \in (300, 400, 500)$ while keeping all other

Double DQN and PER settings fixed. The goal is to examine how sensitive the learned policy and its routing cost are to reasonable changes in the lateness penalty and to verify that the configuration used in the main experiments is sufficiently representative. Table 3 summarizes the mean best-route cost and greedy cost, together with their standard deviations, for each configuration across three random seeds. The results show that differences in β_late produce only minor variations in the total travel time, while the on-time rate remains 100% with no late customers. These findings indicate that the model's performance is relatively stable with respect to moderate changes in β_late, suggesting that the baseline configuration used in the main experiments is sufficiently representative.

Table 3: Sensitivity analysis

Konfigurasi β	Average Best Cost (minutes)	Std Best Cost (minutes)	Average Greedy Cost (minutes)	Std Greedy Cost (minutes)	Mean Number of Late Customers
300	621.57	2.58	650.06	0.00	0
400	616.74	2.46	650.06	0.00	0
500	614.74	1.54	641.61	7.32	0

The parameter β_late serves as the penalty weight for customer service lateness in the reward function. To examine how sensitive the model is to changes in this penalty, we conducted experiments on the 50 customer Friday scenario with three β_{late} values, namely 300, 400, and 500, while keeping all other hyperparameters fixed (learning rate, γ, batch size, network configuration, and exploration–exploitation scheme). For each β_{late} value, the model was run with three random seeds (42, 77, 99), and the table reports the best route (best seed).

Overall, increasing β_late tends to reduce the total travel cost. With β_late = 300, the best cost is about 621.57 minutes. When β_{late} is increased to 400, the best cost decreases to ≈616.74 minutes, and for β_{late} = 500 it further decreases to ≈614.74 minutes. This reduction is only a few minutes (around 0.9% when comparing β_{late} = 300 and 500), indicating that the Double DQN+PER performance is relatively stable with respect to the penalty weight within the tested range. The standard deviation across seeds is also small (around 1.54–2.58 minutes), showing that this trend is consistent and not driven by a single lucky seed.

Importantly, the on-time performance remains 100% for all β_late values. No customers are served late and the average lateness is zero. This implies that, in this scenario, all tested β_late configurations are already large enough to ensure that the model keeps every service within the time window. Consequently, differences in β_late mainly affect the route selection pattern and its interaction with congestion, rather than determining whether customers are late or not.

Overall, these results show that the Double DQN+PER is reasonably robust to variations in β_late within the range 300–500. Increasing β_late from 400 to 500 provides a small improvement in cost and congestion avoidance, but the margin is modest and does not change the main conclusions of the study. Therefore, β_late = 400 can still be used as the baseline configuration in the main experiments, while the results for β_{late} = 300 and 500 are reported as a parameter sensitivity analysis that confirms the model's stability with respect to the lateness penalty weight.

4.5 Performance comparison

This subsection presents a comparative analysis of the performance of the proposed Double Deep Q-Network with Prioritized Experience Replay (Double DQN+PER) against heuristic baselines, namely the Genetic Algorithm (GA) and Ant Colony Optimization (ACO), across different customer sizes and operating days (Monday, Wednesday, and Friday). The comparison focuses on several key indicators: the best route cost (in minutes), the on-time delivery ratio, the number of late customers, and the distribution of road segments that are affected by congestion, successfully avoided, or classified as neutral. Table 4 reports these measurements and provides a quantitative view of the relative advantages of each method in maintaining routing efficiency while satisfying time-window constraints under historical congestion.

Table 4: Performance comparison

Method	Day	Customers	Best Cost (min)	n Late	On-time (%)	Congestion Hit	Congestion Avoided	Neutral Segments
Double DQN+PER	Monday	40	529.4	0	100	10	5	26
Double DQN+PER	Wednesday	40	526.6	0	100	5	7	29
Double DQN+PER	Friday	40	538.7	0	100	2	10	29

Method	Day	Customers	Best Cost (min)	n Late	On-time (%)	Congestion Hit	Congestion Avoided	Neutral Segments
Double DQN+PER	Monday	50	606	0	100	9	9	33
Double DQN+PER	Wednesday	50	595.5	0	100	5	9	37
Double DQN+PER	Friday	50	614.5	0	100	2	12	37
GA	Monday	40	881.83	11	95	5	14	32
GA	Wednesday	40	965.32	11	72.50	4	12	25
GA	Friday	40	922.78	7	82.50	10	3	28
GA	Monday	50	1048.61	18	64	4	14	33
GA	Wednesday	50	1074	18	64	7	9	35
GA	Friday	50	1151.89	21	58	7	10	34
ACO	Monday	40	493.46	0	100	13	2	26
ACO	Wednesday	40	503.48	0	100	9	2	30
ACO	Friday	40	503.09	0	100	9	3	29
ACO	Monday	50	554.19	0	100	15	2	34
ACO	Wednesday	50	584.37	0	100	8	6	37
ACO	Friday	50	556.73	0	100	9	5	37

The comparison results show that ACO consistently yields the lowest route cost (total travel time) across all day scenarios (Monday, Wednesday, Friday) and customer sizes (40 and 50). For 40 customers, ACO produces a total travel time of approximately 493–503 minutes, while for 50 customers it lies in the range of 554–584 minutes. Double DQN+PER ranks second, with slightly longer travel times than ACO, namely around 526–539 minutes for 40 customers and 595–617 minutes for 50 customers, but it remains far more efficient than GA, which yields very high route costs (around 882–965 minutes for 40 customers and 1,049–1,152 minutes for 50 customers). In terms of service level, both Double DQN+PER and ACO maintain schedule performance very well, as there are no late customers (n Late = 0) and the on-time rate reaches 100% in all scenarios, whereas GA fails to satisfy the time-window constraints, with a large number of late customers and a drop in on-time performance to as low as 58% in the worst case (50 customers, Friday).

When viewed from the perspective of interaction with congestion, ACO tends to focus on minimizing total travel time even though the number of segments affected by congestion (Congestion Hit) is relatively higher. In contrast, Double DQN+PER exhibits a more adaptive pattern with respect to congestion, with fewer hits and a substantial number of segments that pass-through hotspots outside peak hours, making it more controlled in dealing with heavy traffic without sacrificing punctuality. GA often appears to "avoid congestion" (with relatively high Congestion Avoided values), but this strategy results in much longer routes, higher total costs, and many violations of time windows. Overall, ACO can be positioned as the classical heuristic that is most efficient in terms of travel time for this instance, while Double DQN+PER serves as a learning-based method that offers an attractive compromise between schedule adherence, congestion management, and flexibility for further deployment in more dynamic and complex distribution scenarios.

4.6 Route and congestion visualization

To provide a clearer picture of the performance of the optimal routes produced by the algorithm, we present visualizations of the distribution paths on a map of Jakarta. The visualization displays the visit sequence of each customer, the depot, and the congestion conditions encountered. Each path is color-coded: red denotes *Congestion Hit* (segments where congestion was encountered), whereas green denotes *Non-hit* segments, covering both *Congestion Avoided* (congested areas successfully bypassed) and *Neutral Segments* (normal traffic). This mapping is consistent with the tabular results and helps readers link numerical metrics such as total travel time and on-time delivery percentage to the actual spatial conditions.

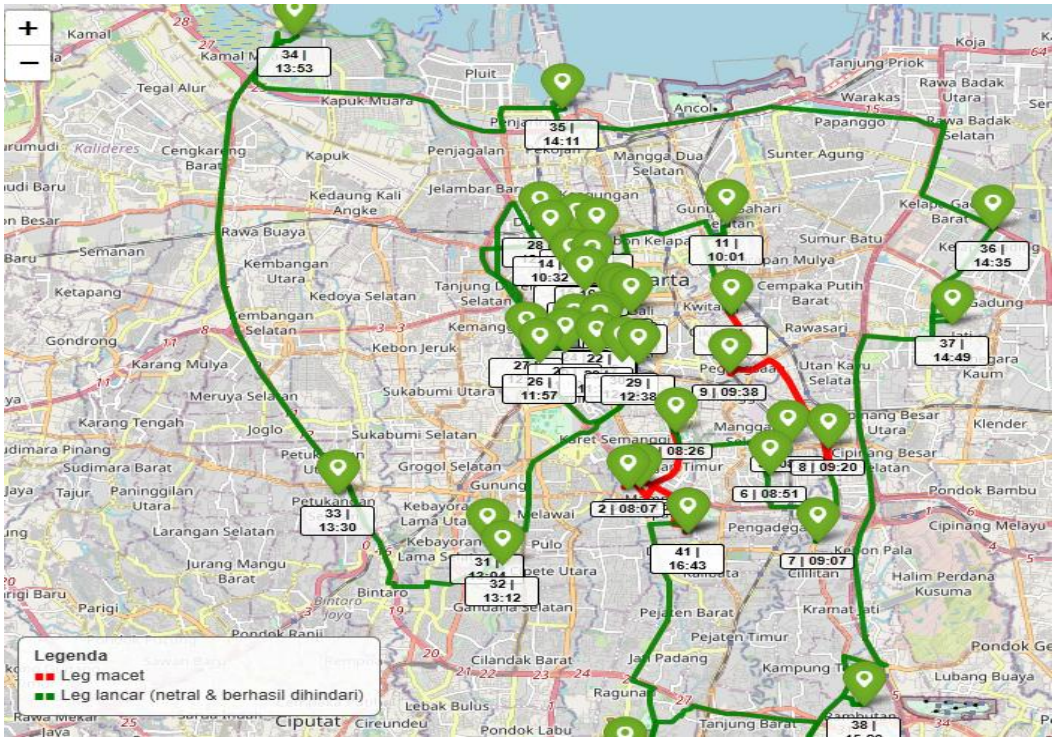

Figure 4: Visualization Results for the 40 Customer Scenario (Monday)

Figure 5: Visualization Results for the 50 Customer Scenario (Monday)

4.7 Discussion

To evaluate the performance of the DQN method enhanced with Double DQN and Prioritized Experience Replay (PER), the model was trained and tested on a Time Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) using Jakarta's historical traffic congestion data. The experiments covered three service days (Monday, Wednesday, and Friday) and two customer scenarios (40 and 50 customers), each run with three different random seeds (42, 77, 99). The best-route summaries show that, for 40 customers, the total time-dependent travel cost lies in the range of approximately 526–539 minutes, while for 50 customers it lies around 595–617 minutes, with 100% on-time delivery and zero late customers in all cases. The per-seed results reported in Table 2 indicate that these outcomes are highly robust with respect to random initialization: for each day and customer size, the three seeds produce almost identical best costs and identical on-time ratios, with only small variations in congestion hits and avoided segments.

The reward and loss curves also exhibit a pattern consistent with reinforcement learning convergence, where episode returns increase while the average training loss decreases and eventually stabilizes. This behavior supports the interpretation that Double DQN effectively reduces overestimation bias in Q-values, leading to more stable learning, while Prioritized Experience Replay accelerates convergence by focusing updates on more informative transitions [24] [25]. In other words, the architectural choices adopted in this study are not only

theoretically motivated but also empirically translated into stable and reproducible routing performance on a time-dependent, congestion-aware TD-VRPTW instance.

The comparison with heuristic methods such as Genetic Algorithm (GA) and Ant Colony Optimization (ACO) provides a clearer picture of the relative position of the proposed DQN approach. As shown in Table 4, ACO consistently achieves the lowest total travel time across all days and customer scenarios. The DQN+PER method ranks second in terms of travel cost, with slightly higher route times than ACO, yet still significantly outperforms GA, which produces much longer routes and a large number of late customers. From a service-level perspective, DQN+PER and ACO both maintain a 100% on-time delivery rate, whereas GA exhibits many time-window violations. When congestion interaction is examined in more detail, DQN+PER shows adaptive behavior: the number of Congestion Hit segments remains at a moderate level, while a substantial number of segments pass through congested areas outside peak hours. This is consistent with previous findings that DQN variants with PER can respond to changing traffic conditions and yield more efficient routes than traditional heuristic methods [26]. The adaptive advantage is also aligned with the Time-Dependent VRP/VRPTW literature, which concludes that modeling travel times based on historical congestion profiles can improve schedule adherence and distribution efficiency [29].

The sensitivity analysis of the lateness penalty parameter β_{late} shows that the main conclusions of this study do not depend on a single hyperparameter setting. Varying β_{late} within a reasonable range leads only to small changes in the best-route cost and does not alter the fact that all customers are served on time. This indicates that the penalty value used is already strong enough to enforce time-window feasibility, while its influence is mainly on fine-grained route selection and congestion-avoidance strategies. Nevertheless, this study has several limitations, including reliance on one week of historical congestion data as a proxy for time-dependent travel times and a focus on a single vehicle and a single depot. Future research could extend the framework to multi-vehicle and multi-depot settings, integrate real-time congestion information and long-term congestion patterns, and conduct broader comparisons with other deep reinforcement learning architectures, thereby providing a more comprehensive view of congestion-aware urban distribution routing in the TD-VRPTW context.

5 Conclusion

This study presented a Deep Q-Network (DQN) model enhanced with Double DQN and Prioritized Experience Replay (PER) for solving the Time-Dependent Vehicle Routing Problem with Time Windows (TD-VRPTW) using historical congestion data from Jakarta. The proposed approach explicitly models time-dependent travel times derived from one week of congestion observations and integrates them into a single-vehicle routing policy for urban distribution. Across three service days (Monday, Wednesday, Friday) and two customer scenarios (40 and 50 customers), each evaluated over three random seeds, the DQN+PER model consistently achieved a 100% on-time delivery rate with zero late customers. The per-seed analysis further showed that the best-route costs and congestion statistics are highly stable across seeds, indicating that the learned policy is robust to stochastic variations in initialization and training.

Comparative experiments against Genetic Algorithm (GA) and Ant Colony Optimization (ACO) highlighted the relative strengths of the proposed method. ACO obtained the lowest total travel time in all scenarios, while the DQN+PER model produced slightly longer routes but still substantially reduced travel time compared with GA, which generated much longer routes and many time-window violations. Both DQN+PER and ACO maintained perfect schedule adherence, whereas GA's on-time performance dropped to as low as 58% in the most congested cases. These results suggest that the proposed DQN+PER model offers a competitive trade-off between travel-time efficiency, strict time-window satisfaction, and congestion-aware routing in a realistic TD-VRPTW setting. Sensitivity analysis on the lateness penalty also indicated that the main conclusions are not dependent on a single hyperparameter choice. Future work will extend this framework to multi-vehicle and multi-depot settings, incorporate real-time traffic and longer-term congestion patterns, and explore additional deep reinforcement learning architectures, thereby providing a more comprehensive foundation for congestion-aware urban distribution planning.

Nevertheless, this study has several limitations, including reliance on historical congestion data as a proxy for real-time conditions and a focus on a single vehicle type. For future research, the model can be extended to incorporate real-time traffic data, multi-vehicle fleets, and the integration of weather and dynamic demand data to better approximate real-world operational conditions.

Acknowledgement

We would like to express our heartfelt gratitude to Universitas Gunadarma for providing the necessary support and resources throughout this research. Our sincere appreciation also goes to the Ministry of Education, Culture, Research, and Technology of Indonesia under the Fundamental Research-Regular, Contact Number: 0419/C3/DT.05.00/2025, for its financial assistance, which has been instrumental in the successful completion of this project. The collaboration and encouragement received from both institutions have greatly contributed to the advancement of our work. Thank you for believing in our vision and enabling us to pursue our research goals.

References

[1] B. Lin, B. Ghaddar, and J. J. I. T. o. I. T. S. Nathwani, "Deep reinforcement learning for the electric vehicle routing problem with time windows," vol. 23, no. 8, pp. 11528-11538, 2021. 10.1109/TITS.2021.3105232

[2] X. Zhang, Y. Yang, J. Cai, Q. Zhu, W. Chen, and Q. Lin, "Deep Reinforcement Learning-Based Multi-Agent Algorithm for Vehicle Routing Problem in Complex Logistics Scenarios," in *2024 International Joint Conference on Neural Networks (IJCNN)*, 2024, pp. 1-8: IEEE. 10.1109/IJCNN60899.2024.10650335

[3] W. Pan and S. Q. J. A. I. Liu, "Deep reinforcement learning for the dynamic and uncertain vehicle routing problem," vol. 53, no. 1, pp. 405-422, 2023. https://doi.org/10.1007/s10489-022-03456-w

[4] F. Guo, Q. Wei, M. Wang, Z. Guo, S. W. J. T. r. p. E. l. Wallace, and t. review, "Deep attention models with dimension-reduction and gate mechanisms for solving practical time-dependent vehicle routing problems," vol. 173, p. 103095, 2023. https://doi.org/10.1016/j.tre.2023.103095

[5] H. Ben Ticha, N. Absi, D. Feillet, A. Quilliot, and T. Van Woensel, "The time-dependent vehicle routing problem with time windows and road-network information," in *Operations Research Forum*, 2021, vol. 2, no. 1, p. 4: Springer. https://doi.org/10.1007/s43069-020-00049-6

[6] M. Gmira, M. Gendreau, A. Lodi, and J.-Y. J. E. J. o. O. R. Potvin, "Tabu search for the time-dependent vehicle routing problem with time windows on a road network," vol. 288, no. 1, pp. 129-140,2021. https://doi.org/10.1016/j.ejor.2020.05.041

[7] M. Ammouriova, E. M. Herrera, M. Neroni, A. A. Juan, and J. J. A. S. Faulin, "Solving vehicle routing problems under uncertainty and in dynamic scenarios: From simheuristics to agile optimization," vol. 13, no. 1, p. 101, 2022. https://doi.org/10.3390/app13010101

[8] Y. Yunita, D. Stiawan, and D. P. Rini, "Vehicle Routing Problem with Time Windows using Hybrid Metaheuristic Dragonfly Algorithm and Variable Neighborhood Search: Work on Progress," in *2024 11th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 2024, pp. 521-525: IEEE. 10.1109/EECSI63442.2024.10776058

[9] S. Ara *et al.*, "Vehicle Routing Problem Solving Using Reinforcement Learning," in *2023 26th International Conference on Computer and Information Technology (ICCIT)*, 2023, pp. 1-6: IEEE. 10.1109/ICCIT60459.2023.10441644

[10] A. Gupta, S. Ghosh, and A. Dhara, "Deep reinforcement learning algorithm for fast solutions to vehicle routing problem with time-windows," in *Proceedings of the 5th Joint International Conference on Data Science & Management of Data (9th ACM IKDD CODS and 27th COMAD)*, 2022, pp. 236-240. https://doi.org/10.1145/3493700.349372

[11] S. Kosolsombat and C. Ratanavilisagul, "Applied Deep Reinforcement Learning for Solving the Vehicle Routing Problem with Time Windows," in *2023 8th International Conference on Computational Intelligence and Applications (ICCIA)*, 2023, pp. 21-25: IEEE. 10.1109/ICCIA59741.2023.00012

[12] F. Moreno-Vera, "Performing deep recurrent double q-learning for atari games," in *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, 2019, pp. 1-4: IEEE. DOI: 10.1109/LA-CCI47412.2019.9036763

[13] J. Escobar-Naranjo, G. Caiza, P. Ayala, E. Jordan, C. A. Garcia, and M. V. J. A. S. Garcia, "Autonomous navigation of robots: optimization with DQN," vol. 13, no. 12, p. 7202, 2023. https://doi.org/10.3390/app13127202

[14] Y. J. H. i. S. E. Yang and Technology, "Path planning under high-dimensional input states based on deep q-network," vol. 120, pp. 576-585, 2024. https://doi.org/10.54097/r6fs0580

[15] D. Seng, J. Zhang, X. J. K. T. o. I. Shi, and I. Systems, "Visual Analysis of Deep Q-network," vol. 15, no. 3, 2021. 10.3837/tiis.2021.03.003

[16] C. Liu, G. Kou, X. Zhou, Y. Peng, H. Sheng, and F. E. J. K.-B. S. Alsaadi, "Time-dependent vehicle routing problem with time windows of city logistics with a congestion avoidance approach," vol. 188, p. 104813, 2020. https://doi.org/10.1016/j.knosys.2019.06.021

[17] H. Fan, Y. Zhang, P. Tian, Y. Lv, H. J. C. Fan, and O. Research, "Time-dependent multi-depot green vehicle routing problem with time windows considering temporal-spatial distance," vol. 129, p. 105211, 2021. https://doi.org/10.1016/j.cor.2021.105211

[18] L. Wang, S. Gao, K. Wang, T. Li, L. Li, and Z. J. J. o. A. T. Chen, "Time-Dependent Electric Vehicle Routing Problem with Time Windows and Path Flexibility," vol. 2020, no. 1, p. 3030197, 2020. https://doi.org/10.1155/2020/3030197

[19] M. A. Cruz-Chávez, A. Rodríguez-León, R. Rivera-López, and M. H. Cruz-Rosales, "A Grid-Based Genetic Approach to Solving the Vehicle Routing Problem with Time Windows," vol. 9, no. 18, p. 3656, 2019. https://doi.org/10.3390/app9183656

[20] G. Chen, J. Gao, and D. J. E. Chen, "Research on Vehicle Routing Problem with Time Windows Based on Improved Genetic Algorithm and Ant Colony Algorithm," vol. 14, no. 4, 2025. https://doi.org/10.3390/electronics14040647

[21] J. Cai, X. Zhang, Q. Lin, L. Dong, W. Chen, and Z. Ming, "Deep Reinforcement Learning for Solving the Vehicle Routing Problem in Practical Logistics," in *2024 IEEE Congress on Evolutionary Computation (CEC)*, 2024, pp. 1-8: IEEE. 10.1109/CEC60901.2024.10612190

[22] B. Yue, J. Ma, J. Shi, and J. J. I. a. Yang, "A deep reinforcement learning-based adaptive search for solving time-dependent green vehicle routing problem," vol. 12, pp. 33400-33419, 2024. 10.1109/ACCESS.2024.3369474

[23] M. Patil, P. Tambolkar, and S. J. I. I. T. S. Midlam-Mohler, "Optimizing Traffic Routes With Enhanced Double Q-Learning," vol. 19, no. 1, p. e70002, 2025. https://doi.org/10.1049/itr2.70002

[24] Z. Zhu, C. Hu, C. Zhu, Y. Zhu, Y. J. J. o. M. S. Sheng, and Engineering, "An improved dueling deep double-q network based on prioritized experience replay for path planning of unmanned surface vehicles," vol. 9, no. 11, p. 1267, 2021. https://doi.org/10.3390/jmse9111267

[25] Q. Huo, "Multi-objective vehicle path planning based on DQN," in *International Conference on Cloud Computing, Performance Computing, and Deep Learning (CCPCDL 2022)*, 2022, vol. 12287, pp. 351-357: SPIE. https://doi.org/10.1117/12.2640707

[26] Y. Niu, F. Zhu, and P. Zhai, "An autonomous decision-making algorithm for ship collision avoidance based on DDQN with prioritized experience replay," in *2023 7th international conference on transportation information and safety (ICTIS)*, 2023, pp. 1174-1180: IEEE. 10.1109/ICTIS60134.2023.10243882

[27] S. Moon, S. Koo, Y. Lim, and H. J. A. S. Joo, "Routing control optimization for autonomous vehicles in mixed traffic flow based on deep reinforcement learning," vol. 14, no. 5, p. 2214, 2024. https://doi.org/10.3390/app14052214

[28] L. P. A. Sanchez, Y. Shen, M. J. J. o. N. Guo, and C. Applications, "Mdq: A qos-congestion aware deep reinforcement learning approach for multi-path routing in sdn," vol. 235, p. 104082, 2025. https://doi.org/10.1016/j.jnca.2024.104082

[29] T. Carić, J. J. P.-T. Fosin, and Transportation, "Using congestion zones for solving the time dependent vehicle routing problem," vol. 32, no. 1, pp. 25-38, 2020. https://doi.org/10.7307/ptt.v32i1.3296