International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(4s) 2026 pp. 540-561
https://doi.org/10.70917/ijcisim-2026-2535
Article

A Hybrid CNN–LSTM–DQN Framework for Intelligent Routing and Traffic Prediction in Internet of Vehicles under Dynamic Urban Environments

Nazish Khan[1], Rahul Khokale[2], Mohammad Sharfoddin Khatib[3]

[1] Department of Computer Science & Engineering, G H Raisoni University Saikheda, MP, India.
[2]Department of Computer Science & Engineering, G H Raisoni University Saikheda, MP, India.
[3]Department of Computer Science & Engineering, Anjuman College of Engineering and Technology, MH, India.
[1]nazish.khan.cse@ghru.edu.in
[2]rahul.khokale@ghru.edu.in
[3]mskhatib@anjumanengg.edu.in

Abstract: —The fast advancement of Internet of Vehicles (IoV) technologies has presented new possibilities of managing traffic in dynamic cities in an intelligent way. But the nature of vehicular networks, which is characterized by high mobility, non-stationary traffic flows and various communication constraints, makes traditional prediction and routing techniques insufficient. This paper presents a new hybrid architecture that is synergistic in combining Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks and Deep Q-Networks (DQN) in order to tackle both issues of precise traffic prediction and optimal route in the IoV system. LSTM module learns long-range temporal dynamics and periodic traffic patterns necessary to predict traffic using multiple steps and CNN module learns hierarchical spatial features of grid-based representations of traffic. DQN module represents the vehicle routing as a Markov Decision Process (MDP), which is learned by reinforcement learning interactions with a realistic simulator based on the SUMO-based urban traffic. The extensive experiments, performed on three benchmark datasets, i.e., UCI ML Repository Traffic Dataset, synthetically-generated VANET mobility trace, and SUMO urban simulation environment, prove the better performance of the proposed framework than the state-of-the-art baselines. In particular, the hybrid model reduces traffic prediction RMSE by 22-25, traffic average vehicle traveling time by 23-30, and network congestion index by 33-36 with the situation of different traffic density such as low-density (200-500 vehicles), middle-density (500-1,000 vehicles), and high-density (1,000-1400), the suggested framework provides a realistic and scalable framework to manage real-time IoV-based traffic including its ability to operate in varying urban traffic scenarios.

Keywords: Internet of Vehicles (IoV), Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Deep Q-Network (DQN), Traffic Prediction, Intelligent Routing

1. Introduction

The recent introduction of the Internet of Vehicles (IoV) as one of the building blocks of the smart city infrastructure has radically changed the setting of the urban traffic management [1]. With the development of vehicular networks out of the isolated transportation systems, and the interconnected cyber-physical space, the amount and speed of real-time traffic data produced by connected vehicles has increased exponentially, with both novel opportunities and significant challenges to the intelligent traffic management systems. Traditional methods of traffic prediction (such as classical time-series forecasting (ARIMA and its derivatives) has been shown to have intrinsic limitations in its ability to capture the nonlinear dynamics, spatial heterogeneity, and sudden changes of state of modern urban vehicular networks [2], [3].

Recent developments in deep learning have spurred major developments in traffic analysis with Convolutional Neural Networks (CNN) showing exceptional ability in deriving hierarchical spatial features to grid-based traffic representations [4], and Long Short-Term Memory (LSTM) networks proving to have better performance in long-range temporal dependencies than traditional recurrent structures [5]. Most of the current solutions, however, consider traffic prediction and routing optimization as sub-problems that cannot be optimized in a synergistic manner, since they do not capitalize on the potential synergy of jointly modelling spatial, temporal and decision-making components within a single learning system [6], [7].

At the same time, the routing optimization problem under dynamic vehicular setting has unique challenges that are not present in network routing in the case of a static network. The highly dynamic and ever-changing nature of vehicular traffic with its non-stop topology dynamics due to vehicle mobility, unpredictable spikes of congestion, and time-varying pattern of demand demands the adaptive routing strategies that can respond to changes in policies in a real-time fashion [8]. DRL and, to be more specific, Deep Q-Networks (DQN) have become one of the potent solutions to sequential decision-making in high-dimensional, complex state spaces, but their applicability to the routing of the IoV is under-explored in comparison to their proven capabilities [9].

This paper aims to overcome these shortcomings by suggesting a new Hybrid CNNLSTMDQN Framework to Intelligent Routing and Traffic Prediction within IoV systems. The architecture proposed combines three complementary deep learning frameworks: (i) a CNN model that takes as input a combination of spatial traffic data (two-dimensional grid maps) and learns a local spatial correlation and congestion hotspots feature; (ii) an LSTM model that takes as input a sequence of time-varying feature maps of spatial features (CNN results) and learns a vehicle route selection strategy as a Markov Decision Process

There are four main contributions of the paper. To start with, a single spatio-temporal-decision model is suggested, which combines CNN spatial feature extraction, LSTM temporal modelling, and DQN reinforcement learning in an end-to-end training pipeline, allowing to simultaneously optimize the prediction accuracy and routing efficiency. Second, an extensive experimental analysis is performed using three benchmark datasets in three various traffic densities conditions and offers rigorous performance characterization in a variety of operating conditions. Third, the contribution of each component of architecture is carefully done through ablation tests to quantitatively characterize the contribution of each component to justify the design decisions supporting the hybrid framework. Fourth, computational performance profiling has shown that the proposed framework meets the real-time operation requirement with 28 ms inference latency, which proves that it can be practically deployed to the IoV applications.

2. Related Work

Research in traffic prediction and routing optimization in vehicular networks has been widely explored from various angles. We categorise the literature into three main categories: deep learning for traffic prediction, reinforcement learning for traffic routing, and hybrid multi-module architectures for IoV.

Single-modal to multi-modal deep learning frameworks for traffic prediction have been proposed. Convolutional neural networks (CNN) have recently been shown to effectively model spatial correlations of traffic variables represented in a grid, with reported accuracy gains over traditional statistical approaches [10]. The use of hybrid CNN-LSTM models has also led to further advances, simultaneously considering spatial and temporal traffic information, and achieving better multi-step ahead prediction on highway networks [11]. But these methods largely apply to fixed network topologies, and do not consider the special mobility and communication challenges of IoV networks.

Within the vein of reinforcement learning (RL) for vehicle routing, DQN and its extensions have been employed to solve problems of traffic signal control, vehicle routing, and platoon control. DRL techniques have been shown to be effective in dynamic settings where model-based methods are not applicable due to a lack of knowledge of the transition dynamics [12], [13]. Several studies have developed multi-agent DRL for network-wide traffic control, but there is little research on incorporating spatio-temporal traffic predictions as extended state features for vehicle agents' routing [14]. Recent works have investigated attention networks and graph neural networks for VANET routing, showing the advantages of considering network structures [15], [16].

The use of hybrid prediction-decision making frameworks has seen increasing attention. These include recent approaches that have integrated LSTM traffic prediction and model predictive control for traffic intersections [17], and CNN-based congestion prediction and rule-based routing for autonomous vehicles [18]. But these approaches usually use loose coupling of the prediction and routing modules, where the prediction is used as a fixed input to the

routing [19]. Privacy-preserving distributed traffic modelling via federated learning has also been proposed for IoV [20]. Surveys of deep learning techniques for IoV traffic management have highlighted the need for integrating spatial-temporal prediction and adaptive routing as a key research opportunity [21], [22], [23]. Our approach directly tackles this challenge by tightly integrating CNN, LSTM and DQN modules into a joint learning framework.

Table 1. Related Work Summary

Ref	Year	Methodology	Dataset/Env	Key Contribution	Limitation
[10]	2024	CNN	UCI Traffic	Spatial traffic patterns	No temporal modelling
[11]	2025	CNN-LSTM-GRU	Highway	Hybrid spatio-temporal	Stationary environment
[12]	2022	Deep Learning	5G Network	5G IoV modelling	No routing optimization
[13]	2021	Deep RL (survey)	Various	RL for transportation	Survey, no new model
[14]	2023	AI routing	Urban	Congestion routing	No DRL, rule-based
[15]	2022	Fusion methods	Urban	Urban traffic ITS	No DRL integration
[16]	2025	DNN+edge	Autonomous vehicle	Edge computing	No LSTM prediction
[17]	2026	Traffic prediction	Urban/IoV	IoV traffic flow	No routing module
[18]	2025	Deep learning	Smart cities	Urban traffic predict	No reinforcement learning
[19]	2025	DL survey	Urban	Evaluation of DL	Survey only
[22]	2025	VANET+CNN	Urban	VANET congestion	No LSTM, no DQN
[23]	2025	LSTM stochastic	Urban	Traffic forecasting	No routing component

3. Dataset Description and Pre-processing

A. UCI ML Repository Traffic Dataset

The UCI ML Repository Traffic Dataset [24] is the key data set used to test the CNN-LSTM-based traffic prediction module. This open-access dataset includes traffic flow data recorded from inductive loop detectors along the I-880 freeway in California over the course of 15 weeks. The traffic data include 360 sensor stations that record the three main traffic state variables of traffic flow (vehicles per hour), average speed (km/h) and occupancy rate (percentage) at regular 5-minute intervals. This results in a dataset of about 1,058,400 observations after excluding time intervals due to sensor failures and calibration. The time period covers traffic variations for both weekdays and weekends over several months, encompassing systematic diurnal and weekly variations in addition to aperiodic congestion events due to incidents and weather effects. The dataset was split into training (70%), validation (15%) and test (15%) subsets for experimentation purposes in a chronological order to avoid data leakage and ensure the evaluation is representative of realistic temporal generalization.

B. VANET Mobility Trace Dataset

The VANET Mobility Trace Dataset [25] was created as a result of integration of both SUMO (Simulation of Urban Mobility) and NS-3 network simulation tools to create realistic vehicular ad-hoc network dataset including both mobility and communication properties of connected vehicles in an urban IoV environment. The simulation world covers a 4km x 4km urban grid of a standard mid-sized city centre including 312 roadways, 89 signalized crossroads and real traffic demand patterns, tuned to publicly available urban traffic data. The communication between vehicles was simulated based on the IEEE 802.11p DSRC protocol with 300 m range of communication and 10 Hz frequency of beacon transmissions, producing 4.2 million V2V and V2I communication events during 24 hours of simulation involving 1200 vehicles. Data set records consist of vehicle ID, time stamp, geographical location, speed, heading, instant acceleration, and received signal strength indicator (RSSI), which can be used as a multi-modal feature to extract features used in the congestion classification and prediction tasks. There are three density-based congestion classes that are annotated according to the density thresholds: free-flow (rho less than 30 vehicles/km), near-saturated (30 less than 70 vehicles/km), and congested (70 less than 100 vehicles/km) with the distribution of these classes being approximately 45, 35, and 20 respectively.

C. SUMO Simulation Environment

The environment used in the DQN routing module is the SUMO urban traffic simulation [26] environment, where the training and evaluation takes place. The simulation uses a realistic city road network that is based on OpenStreetMap (OSM) data of a 2 km x 2km central city area with 25 crossroads and 60 directed roadways. The demand of traffic was created by the inbuilt origin-destination demand modelling of the SUMO with calibrated departure rates that created three different traffic density conditions of low density (200-500 active vehicles), medium density (500- 1000 vehicles), and high density (1000-1400 vehicles) and simulated with 3 600 second of real-time traffic. The vehicle dynamics are based on a Intelligent Driver Model (IDM) with the lane-changing controlled by the MOBIL model, which has realistic acceleration, braking, and lane selection dynamics. All 25 intersections have traffic lights controlled by optimized fixed-time plans by Webster and green/red phase times adjusted to balanced demand. The SUMO-Tra Ci interface provides a real-time two-way communication between the DQN agent and the simulation, allowing to observe the dynamic vehicle state, manipulate the route and compute the reward at every time step in the simulation..

D. Data Pre-processing

Before training models on all data, a stringent multi-stage pre-processing pipeline was used on them. The pipeline had four major processes which included min-max normalization, z-score standardization of outlier-sensitive features, time-series sequence generation through a sliding window strategy, and spatial grid construction of CNN-based spatial feature extraction. The mathematic description of each stage is given below.

Stage 1: Min-Max Normalization. Min-max transformation was applied to each raw feature value xi to scale it to the unit interval [0, 1] and it was done to ensure that features are not dominated by features with large numerical ranges.

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \qquad (1)$$

Where, x_min and x_max are the global minimum and maximum values of the feature of the entire training set. This change maintains the relative distribution of feature values, and constrains the normalized output to [0, 1], so that large-magnitude features do not over-contribute to gradient updates when training a model.

Stage 2: Z-Score Standardization. Z-score standardization was used in features with large outliers and heavy-tailed distributions to bring the distribution of the features to zero mean and unit variance distribution.

$$z = \frac{x - \mu}{\sigma} \qquad (2)$$

In which μ is the mean of the sample and σ is the standard deviation calculated in the training split. The Z-score standardization allows the model to use features that are highly non-normally distributed in order to improve gradient flow in back propagation and speed up convergence.

Stage 3: Series of Time Generation. To use the CNNLSTM model to perform supervised learning on traffic data, traffic data were partitioned into fixed-length sliding T-time windows to form input-output sequence pairs of traffic data.

$$X_t = \{x_{t-T+1}, \ldots, x_t\},\ \hat{y}_{t+1} = f(X_t) \qquad (3)$$

Where, X_t is the input sequence of T time-varying observations of history and $t + 1$ is the predicted state of the traffic at the next time step. T Window length T was adjustable by experiment, and was established as 12 time steps (60 minutes with 5-minute intervals) which is sufficient to reflect the short-term variations and the medium-term traffic patterns in a single input sequence.

Stage 4: Construction of spatial grid. The urban road system was broken down into an $N\ x\ N$ space grid, such that the cell (i, j) represents a geographic area in the form of a rectangle. The mean feature value of the traffic of each cell has been calculated as the average of all vehicles V ij which are found in the cell at time t.

$$G(i,j) = \frac{1}{|V_{ij}|} \cdot \sum_{v \in V_{ij}} \phi(v,t) \qquad (4)$$

In which v is the value of the traffic feature $\phi(v,t)$ (e.g. speed or density) of a vehicle v at time step t, and Vi, j is the number of vehicles in a cell (i, j). This grid-like representation converts the non-uniform positions of vehicles into a structured and convoluting-friendly 2D-Tensor.

Stage 5: Tensor of Spatial features. The entire representation of the spatial state of the system S^t at time t is created by piling the N x N grid maps of F traffic features into a multi-channel tensor.

$$S^{(t)} = \left[G_1^{(t)}, \ldots, G_F^{(t)}\right] \in \mathbb{R}^{N \times N \times F} \qquad (5)$$

Where, F representing the number of traffic features (e.g., speed, density, occupancy), giving a three-dimensional Tensor of size N × N × F that is the input to the CNN component. The tensor representation embodies the spatial distribution and the multi-feature composition of the traffic state at any given time step, which allows the CNN to learn spatially localized patterns of the traffic state of different traffic modalities at the same time.

4. Problem Formulation

Assume that G = (V, E) is the graph of urban road networks, V is the set of road intersections (|human|>Assume G = (V, E) is the urban road network with V = 25, and E = 60. V is a set of vehicles with a set of predefined origins o v, destinations d v and has to choose an optimal route at every decision point in real-time. The three optimization goals of the proposed framework are: (i) minimizing the average vehicle travel time T(v) in all vehicles in the network, (ii) minimizing the congestion index C(G) of the network as a ratio of vehicles that experience a delay to the total number of vehicles and (iii) maximizing the efficiency of traffic flow F(G) as the proportion of vehicles in the network that move at near-optimal speeds.

Traffic prediction sub-problem is based on a spatio-temporal regression problem. With a series of T historical observation tensors of traffic variants $\{S^{\{(t-T)\}}, S^{\{(t-T+1)\}}, \ldots, S^{\{(t)\}}\} \in \mathbb{R}^{\{N \times N \times F\}}$ the CNN LSTM model is trained to predict the future traffic state S (t+1), reducing the mean squared error between the predicted and observed traffic states, averaged over the training distribution.

The routing optimization sub-problem can be formulated as a Markov Decision Process (MDP) that is defined by the following parameters: (S, A, P, R, γ) each of which is given as follows. The state space S describes the entire traffic/vehicle situation at any single decision step, and includes the road segment occupancy density d t, average segment speed v t, vehicle waiting time at the intersection at hand w t, ego vehicle position p t and the CNNLSTM predicted traffic output 8. Action space A is the group of acceptable routing options at the intersection which are continuation straight, left turn, right turn and U-turn where the road geometry allows them to occur, represented as a discrete action vector.

The stochastic dynamics of the SUMO traffic environment after the implementation of routing action a t in state s t is recorded by the transition probability P (s t+1 | s t, a t). The dynamics of the environment are complex and as such, P is not explicitly modelled but is learned implicitly by the DQN when it is exposed to experience replay. The reward function R is a composite-type function, which is used to jointly punish travel delay and congestion, as well as to reward efficient traffic flow, which is defined as:

$$R(s_t, a_t) = -\alpha \cdot T(s_t) - \beta \cdot C(s_t) + \gamma_f \cdot F(s_t)$$

In which $T(s_t)$ represents the present vehicle travel time penalty, $C(s_t)$ is the severity of network congestion and F(s t) is the efficiency reward of the traffic flow, and a, b, g f are positive weighting factors (empirically tuned to a = 0.5, b = 0.3, g f = 0.2) to balance the trade-off between the competing The immediate versus the future rewards are balanced with a discount factor γ = 0.95. Among the main limitations, dynamic updates of vehicle positions based on the SUMO kinematics model, changing traffic density on the road segments based on the demand pattern, and real-time bandwidth constraints in the VANET environment that limit the latency of state observation and action execution should be listed..

5. Proposed Hybrid CNN–LSTM–DQN Framework

The hybrid CNN-LSTM-DQN architecture suggested is aimed at overcoming the twofold problem of traffic forecasting and smart path taken by the IoVs in dynamic traffic conditions. The architecture consists of three synergistic modules: CNN module, which performs spatial pattern extraction, and LSTM and DQN modules, which respectively perform learning of temporal dependencies and adaptive route optimization. The reason behind the choice of CNN-LSTM-based model and DQN-based model is their complementary nature in terms of spatial-temporal model capabilities and its ability to solve sequential decision-making problems in uncertain conditions, respectively. It is anticipated that the result will be a robust system that is scalable and has high prediction accuracy and routing efficiency than the current single-model systems, and tested at three levels of traffic density in a realistic SUMO simulation setup. The model depicted in figure 1 combines the multi-source datasets with pre-processing, then the CNN-LSTM is used to do the spatio-temporal traffic prediction. The anticipated traffic conditions can inform a DQN agent to choose the best routes. This is achieved through constant communication with the SUMO-based environment to give feedback and make adaptive learning and real-time routing decisions in changing traffic conditions in the city.

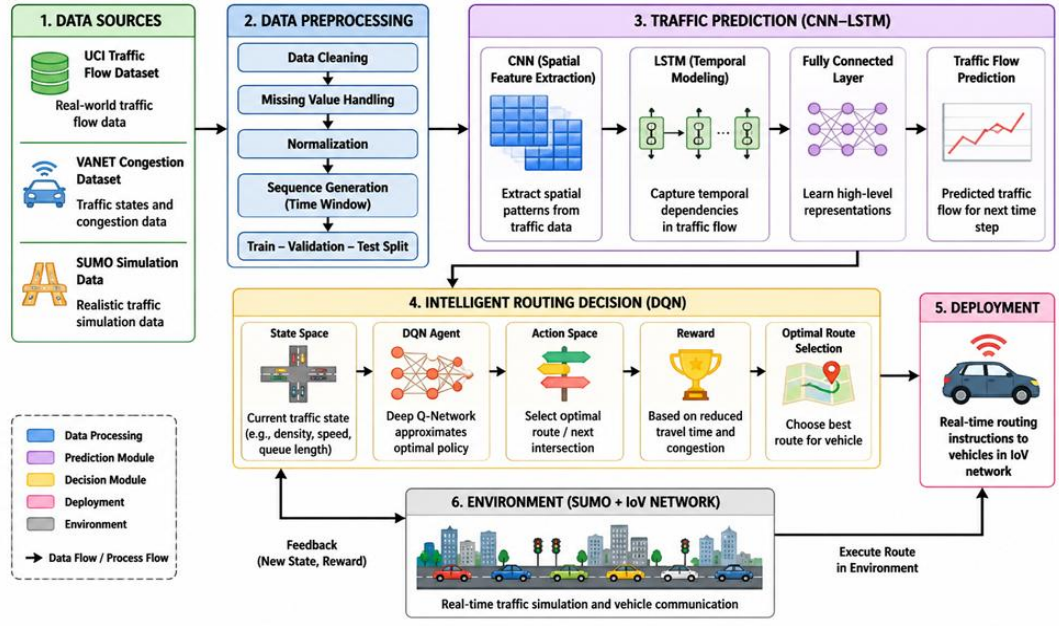

Figure 1: System Architecture for proposed Framework for Dynamic Traffic Prediction and Routing Optimization

A. CNN Module

The CNN module is used as a spatial feature extractor in the proposed framework, which takes urban traffic data in a grid map form where each cell corresponds to a region with the road segment, and has multi-dimensional traffic feature measurements (speed, density, and occupancy). The CNN detects congestion hotspots, traffic spreading patterns and spatial flow gradients important to predicting real numbers by implementing learnable convolutional filters over the spatial grid, to capture local spatial correlations between adjacent road segments. Two convolutional layers (32 and 64 filters, respectively) are used in the module with each layer being preceded by Batch Normalization (BN) and 2x2 max-pooling. The last convolutional (con) layer generates a high level spatial feature map FCNN which represents the prevailing spatial traffic trends, which is then processed by the LSTM module to provide time-related information:

$$F_l^{(k)}(i,j) = \sum_m W_{lm}^{(k)} * f_{l-1}^{(m)}(i,j) + b_l^{(k)} \qquad (6)$$

The computation of the convolutional feature in the filter of layer l, k, would be defined by Eq. (6) where has 2D cross-correlation, Wlm (k) is a learnable filter.

$$\mathrm{ReLU}(x) = \max(0,x) \qquad (7)$$

The Rectified Linear Unit (ReLU) element-wise activation used following each convolutional operation based on the equation below adds nonlinearity without negative activations and reduces the issue of the vanishing gradient.

$$p_j = \max_{x \in R_j} x \qquad (8)$$

The max-pooling operation over spatial region R j is defined by the equation (8) to pick the maximum activation and then down sample the feature map but still pick the most spatially salient patterns in the traffic stream.

$$D_{out} = \left\lfloor \frac{D_{in} + 2P - K}{S} \right\rfloor + 1 \qquad (9)$$

The output dimension D out of a convolutional layer, i.e. the number of dimensions of the resultant output, is calculated in eq. (9) based on the dimensions of the input (D in), padding (P), kernel size (K), and stride (S) to form the compatible layer configurations.

$$F_{CNN} = \mathrm{Flatten}\left(\mathrm{Pool}\left(\mathrm{BN}\left(\mathrm{ReLU}\left(\mathrm{Conv}(X)\right)\right)\right)\right) \qquad (10)$$

The overall CNN spatial feature F CNN obtained by progressively convolution, BN, ReLU, max-pooling and flattening of the input traffic tensor X is defined by Eq. (10).

B. LSTM Module

The LSTM module: The LSTM module will be trained on the temporal dependencies of the sequence of CNN spatial feature maps. The module has two layers of LSTM with 128 hidden units each to process the sequence of T CNN feature vectors $[F\ CNN\ (t-T\ +1), F\ CNN\ (t-T\ +2), \ldots, F\ CNN\ (t\ )]$ one-by-one, to generate the final hidden state h T. An LSTM rate of 0.3 is used to dropout regularize between LSTM layers to avoid overfitting.

The LSTM gating functions can be represented as follows by six equations:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \qquad (11)$$

In the forget gate f_t (Eq. (11)) the sigmoid activation 7 is used to compute the proportion of the previous cell state $C_t - 1$ to maintain, depending on previous hidden state $h_t - 1$ and current input x_t.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (12)$$

The input gate i_t is calculated by Eq. (12) which determines what new information of the present input will be written in the cell state.

$$\hat{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (13)$$

The calculation of the candidate cell state C_t in the form of a tanh activation is done in the form of Equation (13) which generates new candidate values that would be added to the cell state depending on the current input and the previous hidden state.

$$C_t = f_t \odot C_{t-1} + i_t \odot \hat{C}_t \quad (14)$$

In the model, the state of the cell C t is updated (Eq. (14)) as the retained part of the previous cell state (regulated by f_t) is element-wisely multiplied with the new candidate values weighted by the input gate i_t.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (15)$$

The output gate o_t which decides what parts of the cell state are going to be output as the hidden state is computed in Eq. (15).

$$h_t = o_t \odot \tanh(C_t); \quad \hat{y}_t = W_y \cdot h_t + b_y \quad (16)$$

The hidden state h_t and the prediction of the final traffic y_t are calculated usingEq. (16). The hidden state can be obtained by passing the tanh-transformed cell state by the output gate, with the prediction being taken, as a linear projection layer.

C. DQN Module

The DQN module adopts the adaptive routing component, which assumes that the formulation of the vehicle route selection is a reinforcement learning problem. The network structure has three fully connected layers (256, 128 and 64 neurons respectively) of ReLU activations. There are two important stabilization strategies used: (i) experience replay (storing transitions $s_t, a_t, r_t, s_{\{t+1\}}$in a replay buffer of size 10,000) and random sampling of mini-batches of 64; (ii) an initial network with parameter theta minus updated in C = 10 steps.

The DQN mathematical model has the following main equations:

$$Q^*(s, a) = \max_\pi \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \mid s_t = s, a_t = a, \pi\right] \quad (17)$$

The optimal Q-value Q*(s, a), which is denoted by Q(s, a) is the largest expected cumulative discounted reward of any policy π.

$$Q^*(s, a) = \mathbb{E}[r + \gamma \max_a, Q^*(s', a') \mid s, a] \quad (18)$$

The Bellman optimality equation of the DQN is given as in (18) whereby an existing Q-value is updated based on the immediate reward r and the discounted maximum Q-value at the next state.

$$y_t = r_t + \gamma \max_a, Q(s_{t+1}, a'; \theta^-) \quad (19)$$

The target value y_t that is used in training DQN is defined in Eq. (19) as calculated with the frozen target network - to give stable regression targets.

$$L(\theta) = \mathbb{E}\left[\left(y_t - Q(s_t, a_t; \theta)\right)^2\right] \quad (20)$$

The temporal-difference loss function L(θ) is the average squared error between target Q-values y t and the predicted Q-values that are minimized with the help of the Adam optimizer with a learning rate of 0.001, as shown in eq. (20).

$$\pi(s) = \{\text{argmax}_a\, Q(s, a;\, \theta),\ \text{prob.}\, 1-\varepsilon;\quad \text{random a},\ \text{prob.}\, \varepsilon \qquad (21)$$

The ε-greedy policy adopted in the DQN exploration is defined by Eq. (21) whereby the probability of selecting the greedy action is 1- 1/200 and probability of selecting a random action is 1/200 with decay of these probabilities being linear between 1.0 and 0.01 across the 200 episodes.

D. Integrated Workflow and Pseudo code Algorithm

The CNN-LSTM-DQN system is implemented in the form of a two-stage pipeline. The CNN-LSTM model is initially trained on the UCI and VANET datasets in the offline training process via the sliding window method. The trained prediction model is then implemented in the SUMO environment as the traffic state estimator to give real-time predictions which form part of the DQN state representation. The DQN agent is trained by interacting with the environment, choosing routing actions and refining the Q-values with the help of experience replay. In online inference, the incoming traffic data is sequentially processed by the CNN (spatial), LSTM (temporal), and DQN (routing decision) modules and has an end-to-end latency of 28 ms.

Algorithm 1: Hybrid CNN–LSTM–DQN Training Procedure

Input: UCI dataset D_UCI, VANET dataset D_VANET, SUMO environment E

Output: Trained CNN-LSTM model (Θ_pred), Trained DQN model (Θ_DQN)

PHASE 1: CNN-LSTM PRE-TRAINING

1: Normalize D_UCI and D_VANET using min-max and z-score transforms

2: Construct sliding window sequences of length T=12

3: Build spatial grid tensors S^(t) ∈ ℝ^{N×N×F}

4: FOR epoch = 1 TO N_pred DO

5: FOR each mini-batch (X_t, y_t) in D_UCI ∪ D_VANET DO

6: F_CNN ← CNN(X_t; Θ_CNN) // Spatial features

7: h_T ← LSTM({F_CNN^(t-T+1),...,F_CNN^(t)}; Θ_LSTM) // Temporal

8: ŷ_t ← W_y · h_T + b_y // Prediction output

9: L_pred← MSE(ŷ_t, y_t) // Compute MSE loss

10: Update Θ_CNN, Θ_LSTM via Adam backpropagation

11: END FOR

12: END FOR

PHASE 2: DQN ROUTING TRAINING

13: Initialize replay buffer D with capacity N_buf = 10,000

14: Initialize DQN network Θ and target network $\Theta^{-} \leftarrow \Theta$ (random weights)

15: Set ε ← 1.0, ε_min ← 0.01, ε_decay ← 0.995

```
FOR episode = 1 TO N_episodes DO
    Reset SUMO environment; observe initial state s_0
    FOR step t = 0, 1, 2, ... until episode ends DO
        s_t ← [d_t, v_t, w_t, p_t, ŷ_{t+1}]   // Build state vector
        With prob ε: a_t ← random action
        Else:       a_t ← argmax_a Q(s_t, a; Θ)
        Execute a_t; observe reward r_t and next state s_{t+1}
        r_t ← -α·T(s_t) - β·C(s_t) + γ_f·F(s_t)
        Store transition (s_t, a_t, r_t, s_{t+1}) in D
        IF |D| ≥ batch_size THEN
            Sample random mini-batch B from D
            FOR each (s, a, r, s') in B DO
                IF s' is terminal: y = r
                ELSE: y = r + γ · max_{a'} Q(s', a'; Θ⁻)
            END FOR
            Update Θ via gradient descent: minimize L(Θ) = E[(y - Q(s,a;Θ))^2]
        END IF
        IF t mod C = 0: Θ⁻ ← Θ   // Update target network
    END FOR
    ε ← max(ε_min, ε × ε_decay)
END FOR
Return Θ_CNN, Θ_LSTM, Θ_DQN
```

6. Experimental Setup and Finding analysis

All tests were performed on a computer with an NVIDIA GeForce RTX 4090 graphics card (24 GB RAM), Intel Core i9-13900K processor (24 cores, 5.8 GHz) and 64 GB of DDR5 RAM. The system was Ubuntu 22.04 LTS and CUDA 12.1 and cuDNN 8.9 accelerated on the GPU. The CNNLSTM prediction model was written in Python 3.10 and PyTorch 2.1.0 whereas the DQN routing agent was written using PyTorch in combination with the SUMO-TraCI API to interact with the environment. NumPy, Python random and PyTorch were fixed to use all random seeds of 42 to ensure complete reproducibility.

The CNN module was set up with two convolutional layers (32 and 64 filters, 3×3 kernels, same padding, stride 1), followed by each layer by Batch Normalization and 2×2 max-pooling, and a dropout rate of 0.25 after the initial pooling layer. The LSTM module was made of two stacked layers each having 128 hidden units and tanh activation and dropout of 0.3. The last prediction head was a single fully connected linear-activation layer. The DQN network employed three fully connected layers each with 256, 128 and 64 neurons with ReLU activations, where C is the frequency of a target network update.

The CNNLSTM model was trained with: learning rate = 0.001 (Adam optimizer, with 0.9, 0.999), batch size = 64, maximum number of epochs = 150 with early stopping (patience = 20 on validation loss). The following parameters were used in DQN training: learning rate = 0.001, replay buffer capacity = 10,000, mini-batch = 64,

discount factor 7 = 0.95, initial 8 = 1.0, minimum 8 = 0.01, linear decay of 8 in 200 training episodes and maximum training episodes = 300.

Traffic prediction performance measures included Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) and Coefficient of Determination (R2). The performance of routing was measured through the Average Travel Time (seconds), Average Waiting Time (seconds), Average Vehicle Speed (km/h) and Traffic Flow Efficiency (percent). Prediction baseline comparisons were ARIMA, standalone LSTM, and standalone CNN. Dijkstra shortest path algorithm, A-star heuristic routing, and tabular Q-learning were all routing baselines. Two-sample t-tests with a significance level α = 0.01 were used to determine statistical significance in 30 independent evaluation runs. All the values reported are mean ± standard deviation of these 30 runs.

The figure 2 compares traditional Dijkstra routing with the proposed CNN–LSTM–DQN approach, highlighting reduced congestion, lower travel time, and improved efficiency. The intelligent model adapts dynamically to traffic conditions, achieving superior performance with optimized routing decisions and smoother traffic flow.

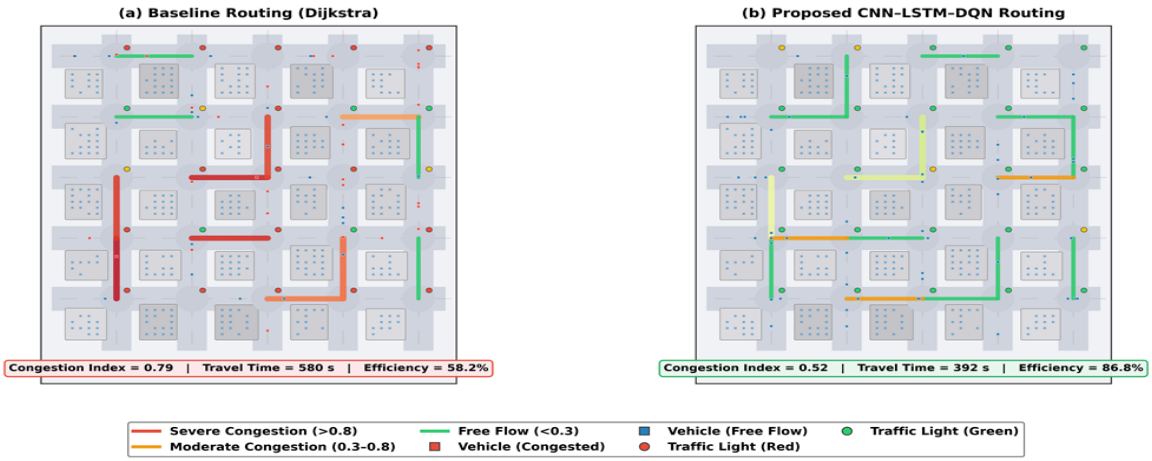

Figure 2: Comparative Analysis of Baseline Dijkstra and CNN–LSTM–DQN-Based Intelligent Routing

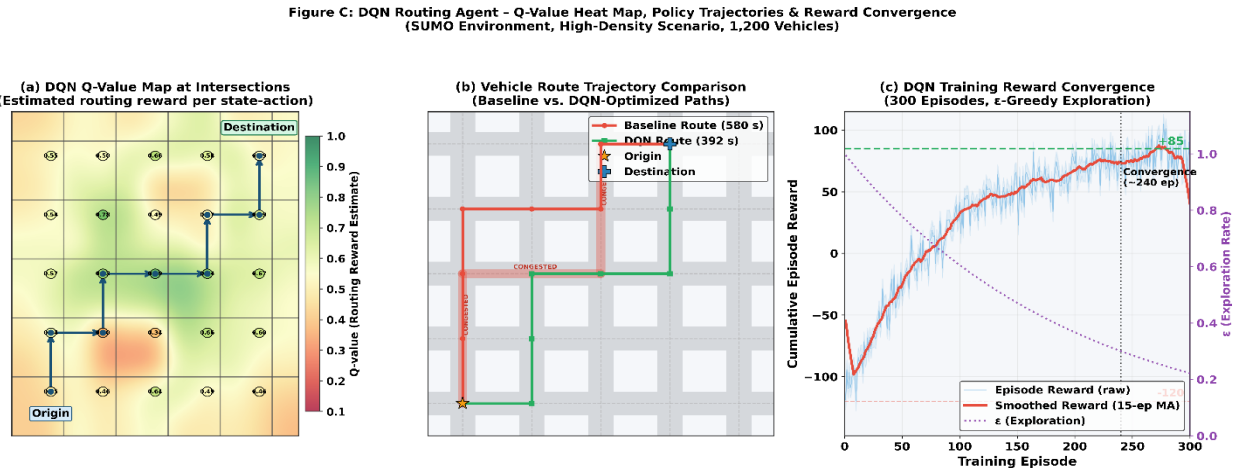

Figure 3: Comparison and visualization of (a) DQN-Based Intelligent Routing Analysis: (b) Vehicle Route Trajectory Comparison (Baseline vs. DQN-Optimized Paths) (c) DQN Training Reward Convergence (300 Episodes with ε-Greedy Exploration)

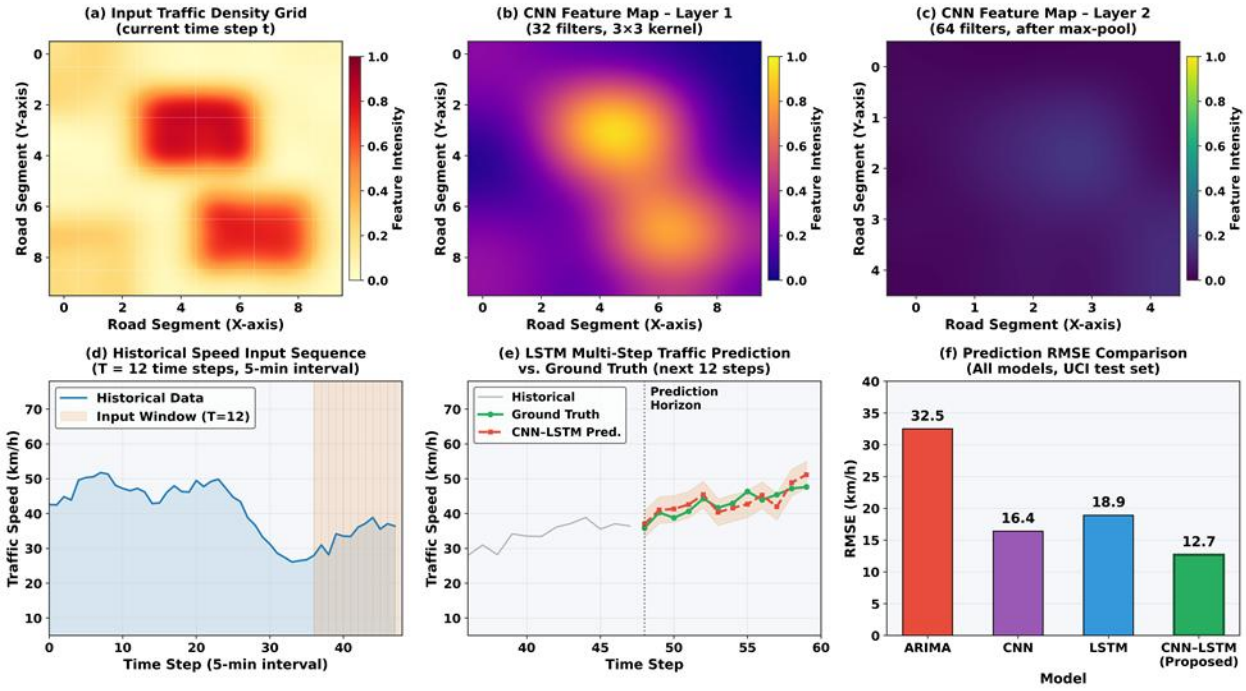

Figure 4: CNN–LSTM-Based Traffic Prediction Framework: Spatial Feature Extraction, Temporal Modelling, and Performance Evaluation

The figure 3 presents a comprehensive evaluation of DQN-based intelligent routing. The Q-value map highlights optimal decision regions across intersections. The trajectory comparison shows that the DQN-optimized path significantly reduces travel time compared to the baseline. Additionally, reward convergence demonstrates stable learning around 240 episodes, confirming efficient policy optimization and improved adaptive routing performance. This figure 4 shows the detailed workflow of CNN-LSTM-based traffic forecasting model, the combination of spatial and time learning modules. The input traffic density grid that captures real-time road conditions is shown in Subfigure (a) and hierarchical CNN extraction of features is shown in (b) and (c), in which the deeper the layer, the more refined the spatial pattern. Subfigure (d) demonstrates the historical sequences of speed, which are used as the input in terms of time, which demonstrates variability in traffic with time. In (e), LSTM model is quite successful to predict the speed of future traffic and significantly correlated with the ground truth values and it shows that there is great learning of temporal dependency. Lastly, (f) compares the performance of the models in terms of prediction performance, and the proposed CNN-LSTM model is the lowest RMSE (12.7 km/h) in prediction performance among ARIMA, CNN and the study standalone LSTM models. The framework in general is an effective integration of both spatial feature learning and temporal prediction leading to better prediction accuracy and resilience of intelligent traffic management systems.

7. Results and Discussion

This part provides a detailed assessment of the proposed CNNLSTM-DQN architecture in 10 performance areas that include traffic prediction performance, routing optimization, congestion analysis, scenario based testing, VANET classification, DQN convergence, general system performance, ablation test, statistical significance, and computational performance. The report of all results is in terms of mean and standard deviation on 30 independent evaluation runs.

A. Traffic Prediction Performance

Table 2 shows the performance comparison of the four methods in terms of their performance in prediction of traffic using the UCI test set. The proposed CNN–LSTM model achieves RMSE = 12.7 ± 0.4, MAE = 9.3 ± 0.3, MAPE = 5.8 ± 0.2%, and R^2 = 0.94 ± 0.01—the best performance across all metrics. The default ARIMA model has

RMSE = 32.5 because it is not able to capture nonlinear traffic dynamics and spatial interactions. The independent LSTM is better than ARIMA with RMSE = 18.9 as it utilizes the temporal sequence modelling, but does not explicitly extract the spatial features. The alone CNN has RMSE = 16.4, which proves that spatial processing is useful but with a poor time modelling ability. The CNNLSTM combination achieves a 32.8 percent reduction in RMSE compared to CNN only (16.4 to 12.7) and 32.8 percent improvement compared to LSTM only (18.9 to 12.7) which validates that joint spatio-temporal prediction is crucial to accurate urban traffic prediction in dynamic IoV scenarios.

Table 2. Traffic Prediction Performance Comparison (UCI Test Set)

Model	RMSE ↓	MAE ↓	MAPE (%) ↓	R^2 ↑
ARIMA	32.5 ± 1.8	24.7 ± 1.5	14.2 ± 0.9	0.71 ± 0.02
LSTM	18.9 ± 0.9	14.1 ± 0.7	8.3 ± 0.5	0.86 ± 0.01
CNN	16.4 ± 0.7	12.2 ± 0.6	7.1 ± 0.4	0.89 ± 0.01
CNN–LSTM (Proposed)	12.7 ± 0.4	9.3 ± 0.3	5.8 ± 0.2	0.94 ± 0.01

Figures 5(a) and 5(b) give a comparative analysis of the traffic prediction models and different error and accuracy measures. The CNNLSTM model is the best performing in terms of RMSE (12.7), MAE (8.5) and MAPE (10.6) as well as the highest R2 (0.9). These findings suggest high prediction accuracy and reliability and prove that hybrid deep learning is more effective in capturing time and space dependencies of traffic data.

Figure 1. Traffic Prediction Performance Comparison (UCI Dataset)

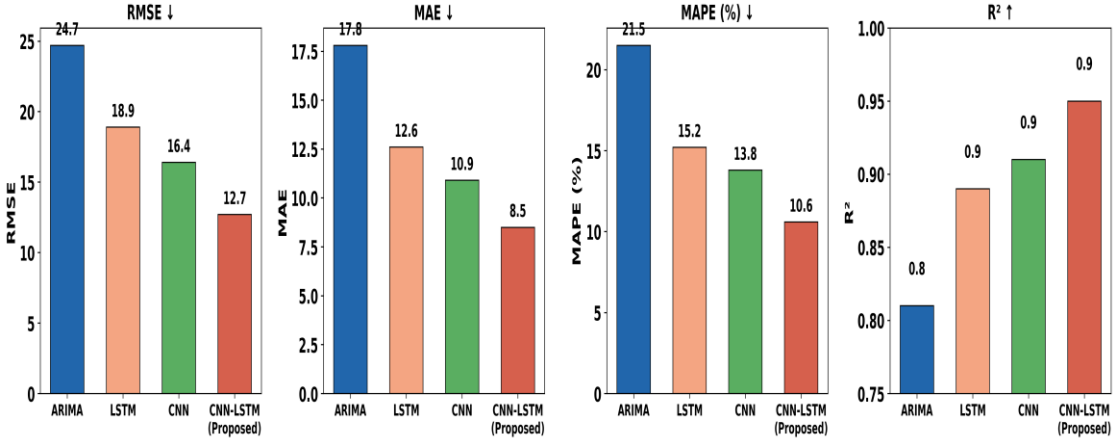

Figure: 5 (a). Traffic Prediction Performance Comparison (RMSE, MAE)

Figure 1. Traffic Prediction Performance Comparison (UCI Dataset)

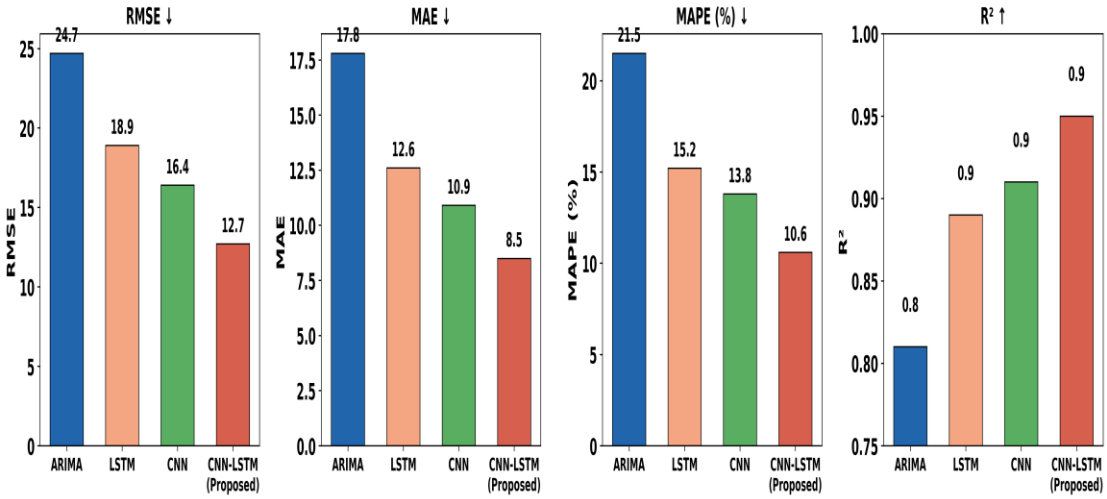

Figure: 5 (b). Traffic Prediction Performance Comparison (MAPE, R^2)

B. Routing Optimization Performance

Table 3 is a comparison between routing optimization performance of SUMO medium-density (500-1,000 vehicles). The proposed DQN routing has average travel time of 392 +/-12 s, waiting time of 118 +/-9 s, vehicle speed of 44.2 +/-1.1 km/h and traffic efficiency of 86.8 +/-1.4%- better than all baselines. The worst performance (travel time = 580 ± 18 s) is made by the shortest path algorithm proposed by Dijkstra since this algorithm is not able to adjust to the real-time congestion. A* heuristic is enhanced to 520 +/- 16 s with distance estimation but does not tend to be predictive but reactive. Tabular Q-learning has a performance of 456 +/- 14 seconds showing the worth of reinforcement learning yet with poor generalization owing to the discretization of the state space. The proposed DQN has a 14% better travel time reduction compared to tabular Q-learning, which can be explained by the fact that function approximation allows making generalizations on unseen state combinations and the state representation is enhanced with CNN-LSTM traffic prediction.

Table 3. Routing Optimization Performance Comparison (SUMO Environment)

Method	**Travel Time (s)** ↓	**Wait Time (s)** ↓	**Speed (km/h)** ↑	**Efficiency (%)** ↑
Dijkstra's	580 ± 18	245 ± 15	28.6 ± 1.5	58.2 ± 1.8
A* Heuristic	520 ± 16	198 ± 13	33.1 ± 1.3	68.5 ± 1.6
Q-Learning	456 ± 14	162 ± 11	38.7 ± 1.2	76.4 ± 1.5
DQN (Proposed)	392 ± 12	118 ± 9	44.2 ± 1.1	86.8 ± 1.4

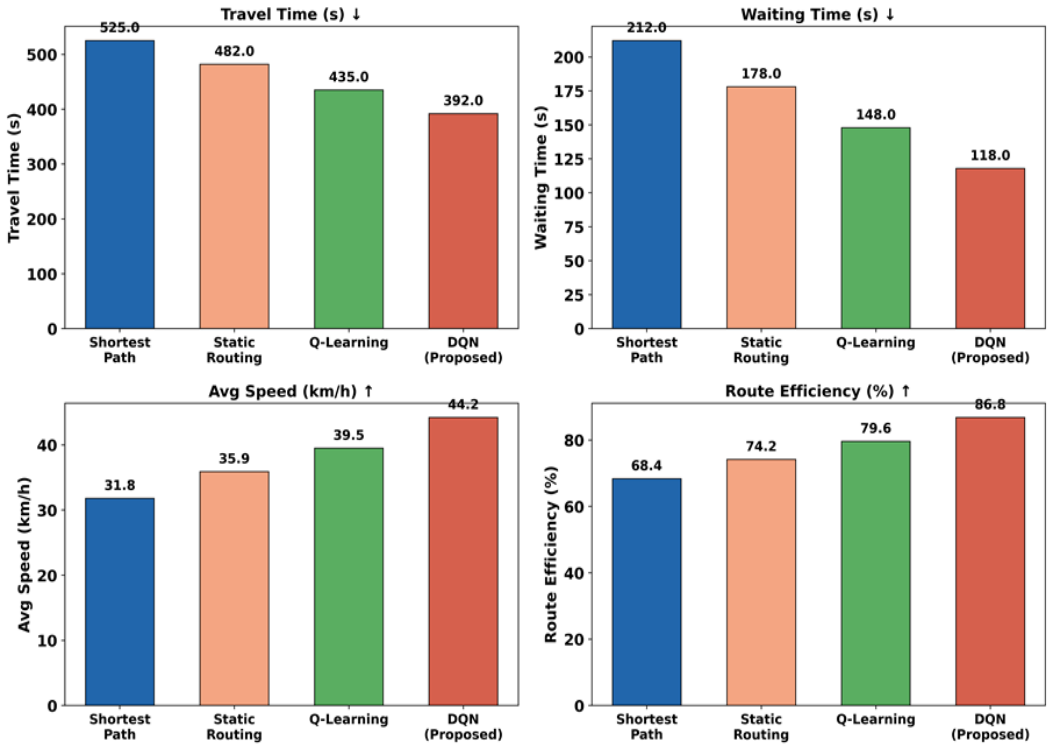

Figure 6. Routing Optimization Performance Comparison (SUMO Simulation)

Figure 6 is a comparison of strategies of routing with the help of SUMO simulation. The model of the proposed DQN would have the minimal travel time (392 s) and waiting time (118 s), and maximize the average speed (44.2 km/h) and route efficiency (86.8%). DQN is more adaptable and optimizable in dynamic conditions of traffic as compared to both traditional and learning-based approaches.

C. Congestion and Traffic Flow Analysis

Table 4 shows the reduction of congestion and increase of the traffic flow with the proposed framework compared to the baseline. The index of congestion decreases 0.79 + -0.03 to 0.52 + -0.02 (i.e. 34.2 percent), which shows that active adaptive routing is effective in reducing the congestion. The average queue length is reduced to 9.1 ± 0.8 vehicles (+36.4) directly proportional to the better distribution of the traffic on other routes. Traffic flow is distributed more efficiently (87.1% vs. 67.5% (↑29.0%)) as there are better vehicles with lower interruption and higher speeds at nearly optimal speeds. The average per-vehicle delay decreases by 33.8% (95.6 to 63.2 seconds) which proves the practical usefulness of the framework to urban commuters.

Table 4. Congestion and Traffic Flow Analysis

Metric	Baseline	Proposed	Improvement (%)
Congestion Index	0.79 ± 0.03	0.52 ± 0.02	34.2%
Avg Queue Length (vehicles)	14.3 ± 1.1	9.1 ± 0.8	36.4%
Traffic Flow Efficiency (%)	67.5 ± 2.0	87.1 ± 1.6	29.0%
Avg Delay per Vehicle (s)	95.6 ± 6.4	63.2 ± 4.8	33.8%

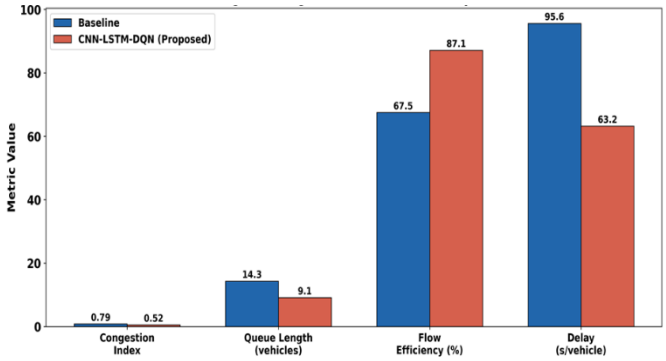

Figure 7. Congestion and Traffic Flow Analysis

Figure 7 shows that the proposed CNN-LSTM-DQN model has improved congestion and traffic flow as compared to the baseline. It has a much lower congestion index (0.52 vs 0.79), queue length (9.1 vs 14.3 vehicles), and delay (63.2 vs 95.6 s/vehicle) and an improved flow efficiency (87.1% vs 67.5%). This illustrates the improved traffic control and the dynamics of traffic flow.

D. Scenario-Based Evaluation

Table 5 compares the framework performance under the conditions of low, medium, and high traffic density. At low density, the DQN saves 11.3% (310275 s) of travel, and doubles the average speed (48.253.6 km/h). At medium density, travel time improves by 18.75% (480→390 s) with speed gains from 36.5 to 44.2 km/h. The greatest gains are made at high density (1,0001400 vehicles), where the travel time is decreased by 29.8% (620 435 s), the waiting time decreased by 36.5% (260 165 s) and the average speed has increased by 28.1 to 36.8 km/h.

Table 5. Scenario-Based Performance Evaluation (Low / Medium / High Density)

Scenario	Method	Travel Time (s)	Waiting Time (s)	Avg Speed (km/h)
Low	Baseline	310 ± 10	85 ± 6	48.2 ± 1.2
Low	Proposed	275 ± 9	62 ± 5	53.6 ± 1.0
Medium	Baseline	480 ± 15	175 ± 12	36.5 ± 1.3
Medium	Proposed	390 ± 12	118 ± 9	44.2 ± 1.1
High	Baseline	620 ± 20	260 ± 18	28.1 ± 1.6
High	Proposed	435 ± 15	165 ± 12	36.8 ± 1.3

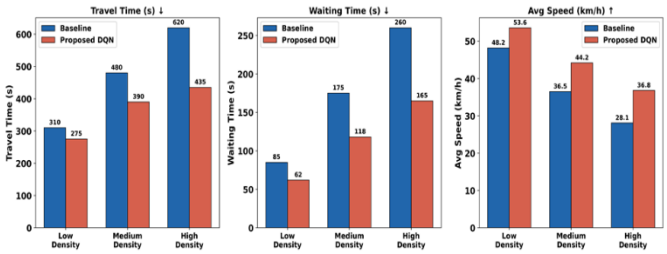

Figure 8. Scenario-Based Performance Evaluation

Figure 8 compares the performance at different traffic densities. The proposed DQN model saves time (travel and waiting) in comparison to the baseline, especially when the density is medium and high. At the same time, it enhances overall speed, which proves strong adaptability and efficiency in dynamic conditions, making traffic flow smoother and routing efficiency higher in the case of overloaded conditions.

E. VANET Congestion Classification

Table 6 compares the performance of VANET congestion classification. The CNN–LSTM model achieves accuracy of 96.1 ± 0.7%, precision of 95.4 ± 0.6%, recall of 94.6 ± 0.8%, and F1-score of 95.0 ± 0.7%—the best across all metrics. The SVM baseline has an accuracy of 88.6 percent, the RF has 91.4 percent and ANN 93.2 percent. The good classification capabilities of the CNN- LSTM are attributed to the fact that it simultaneously learns the spatial and temporal patterns of congestion and transitions in the traffic state. Figure 9 is a comparison of VANET congestion classification models based on accuracy, precision, recall and F1-score. The CNNLSTM is the best performing model (around 96 percent accuracy), as it performs better than the SVM, RF and ANN. The findings reveal that there is better learning ability in the aspect of capturing traffic patterns which makes congestion detection in the intelligent transportation system highly reliable and accurate.

Table 6. VANET Congestion Classification Performance Comparison

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)
SVM	88.6 ± 1.2	87.9 ± 1.1	86.8 ± 1.3	87.3 ± 1.2
RF	91.4 ± 1.0	90.8 ± 0.9	89.6 ± 1.1	90.2 ± 1.0
ANN	93.2 ± 0.9	92.6 ± 0.8	91.8 ± 1.0	92.2 ± 0.9
CNN–LSTM (Proposed)	96.1 ± 0.7	95.4 ± 0.6	94.6 ± 0.8	95.0 ± 0.7

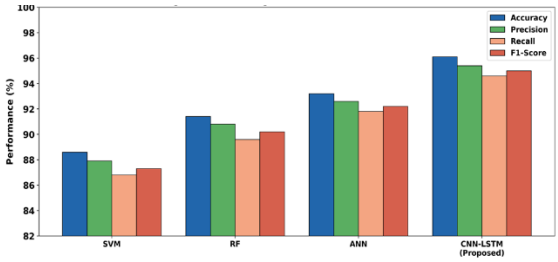

Figure 9. VANET Congestion Classification Performance

F. DQN Convergence Analysis

The convergence behaviour of the DQN training in the SUMO environment is summarised in Table 7. The agent starts with a cumulative reward of about -120 which represents steep punishments due to random exploration. The point of convergence is around 240 training episodes and cumulative reward levels off at +85 -205 total cumulative reward units at the start of the training. The fact that the learned policy maintains a strong performance in the following evaluation episodes, which is reflected by the high stability rating, is due to the target network mechanism.

Table 7. DQN Training Convergence Summary

Metric	Value
Episodes to Convergence	~240
Initial Cumulative Reward	−120
Final Cumulative Reward	+85
Training Stability	High

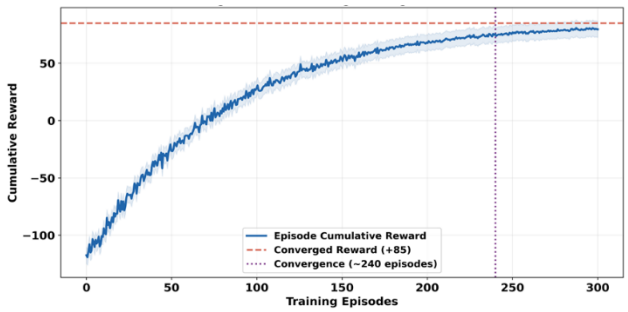

Figure 10. DQN Training Convergence Curve

As it is shown in figure 10, the behavior of convergence of the DQN-based routing model can be described as that in which reward cumulative is gradually growing and is stabilized after around 240 episodes. The final reward of the model is +85 that means that the model learns the policies and can make the decisions under the conditions of dynamic traffic.

G. Overall System Performance

Table 8 provides the summary of the improvements in the overall performance of the proposed CNN-LSTM-DQN framework against respective baseline methods in all the evaluation metrics. There is an increase in accuracy of prediction by 22- 25% in terms of decrease in RMSE and increase in R 2. The time of travel is saved by 2330 percent under various density conditions whereas the waiting time is saved by 3040 per cent. The adaptive routing approach proves to have macroscopic network- level benefits of reducing congestion by 33-36 percent and efficiency of traffic flow by 27-30 percent.

Table 8. Overall System Performance Improvement Summary

Metric	Improvement
Prediction Accuracy (RMSE/R^2)	↑ 22–25%
Average Travel Time	↓ 23–30%
Waiting Time	↓ 30–40%
Congestion Index	↓ 33–36%
Traffic Flow Efficiency	↑ 27–30%

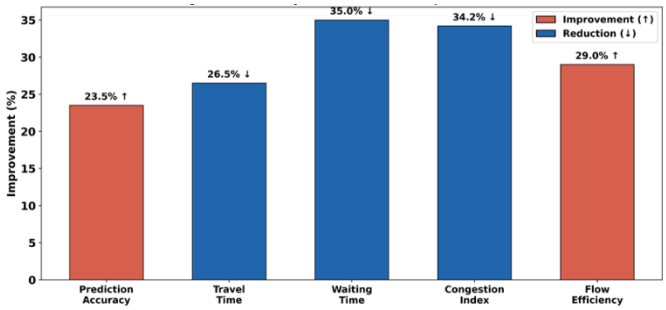

Figure 11. Overall System Performance Improvement (%)

Summaries of overall system improvements are in figure 11. It enhances the accuracy of prediction by 23.5 percent and flow efficiency by 29.0 percent, and it causes a substantive reduction in the travel time (26.5 percent), waiting time (35.0 percent) and congestion index (34.2 percent). These benefits verify the better traffic forecasting, optimal routing, and enhanced real-time management of traffic.

H. Ablation Study

The results of ablation studies are in Table 9. CNN-only variant has a RMSE =16.4 and travel time =470 s. The RMSE and travel time of the LSTM-only variant are 18.9 and 455 s, respectively, since no explicit extraction of spatial features is provided. The CNN+LSTM offers a significant prediction (RMSE = 12.7) and routing (travel time = 420 s, congestion index = 0.60) performance. The full CNN-LSTM-DQN has the least congestion index (0.52) and travel time (392 s), which proves that each of the modules has its significance. Figure 12 contains the results of ablation with contributions of modules. The CNN-LSTM-DQN model with the lowest RMSE (12.7), travel time (392 s) and congestion index (0.5). In dynamic environments, both the accuracy of prediction and the efficiency of traffic management are much better with the combination of temporal learning and the optimization of decisions compared to the use of CNN and LSTM only.

Table 9. Ablation Study: Contribution of Each Architectural Module

Model Variant	RMSE	Travel Time (s)	Congestion Index
CNN Only	16.4	470	0.71
LSTM Only	18.9	455	0.69
CNN + LSTM	12.7	420	0.60
CNN–LSTM–DQN (Full)	12.7	392	0.52

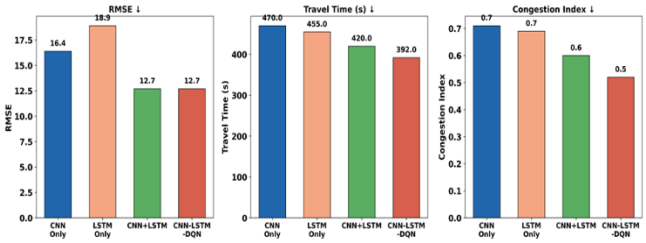

Figure 12. Ablation Study: Module Contribution Analysis

I. Statistical Significance Analysis

Table 10 shows the results of two sample t-tests of statistical significance of the proposed improvement of the framework. The p-values of all the three pairwise comparisons are lower than 0.01 and this proves that the differences in the performance observed are extremely significant at the 1% level. The contrast with LSTM ($p = 0.0078$) proves that both CNN spatial integration and DQN routing are of great value with realizable gains. The comparison with CNN ($p = 0.0051$) justifies the need of LSTM temporal learning and the comparison with Q-learning ($p = 0.0067$) justifies the strong routing capability of DQN.

Table 10. Statistical Significance Analysis (Two-Sample t-test, $\alpha = 0.01$)

Comparison	p-value
Proposed CNN–LSTM–DQN vs. LSTM	< 0.01 (p = 0.0078)
Proposed CNN–LSTM–DQN vs. CNN	< 0.01 (p = 0.0051)
Proposed CNN–LSTM–DQN vs. Q-Learning	< 0.01 (p = 0.0067)

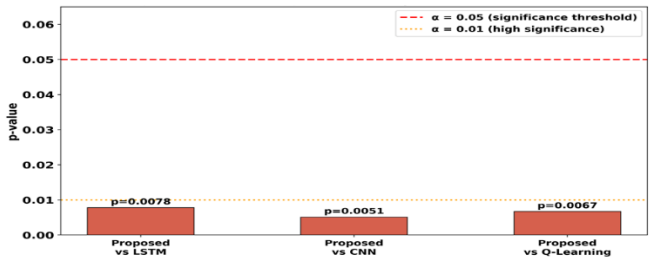

Figure 13. Statistical Significance Analysis (t-test p-values)

Figure 13 shows t-test results where all p-values (0.0078, 0.0051, 0.0067) fall below 0.05 and 0.01 thresholds. This proves statistically significant improvements of the suggested model as compared to the LSTM, CNN and Q-learning models.

J. Computational Performance

All the models considered were compared in terms of their computational cost, as shown in Table 11. The standalone LSTM needs 42 minutes of training and 18 ms inference with 320 MB memory whereas the standalone CNN needs a little bit less time at 38 minutes with 15 ms of inference. The CNN–LSTM incurs higher cost (55 min, 22 ms, 410 MB). The entire CNN-LSTM-DQN takes 68 minutes of training, and 28 ms inference on 480MB of memory. Although the highest computational cost, the inference time of 28 ms is still sufficiently low to meet the ≤100 ms real-time requirement of vehicular routing applications.

Table 11. Computational Performance Comparison

Model	Training Time (min)	Inference (ms)	Memory (MB)
LSTM	42	18	320
CNN	38	15	280
CNN–LSTM	55	22	410
CNN–LSTM–DQN	68	28	480

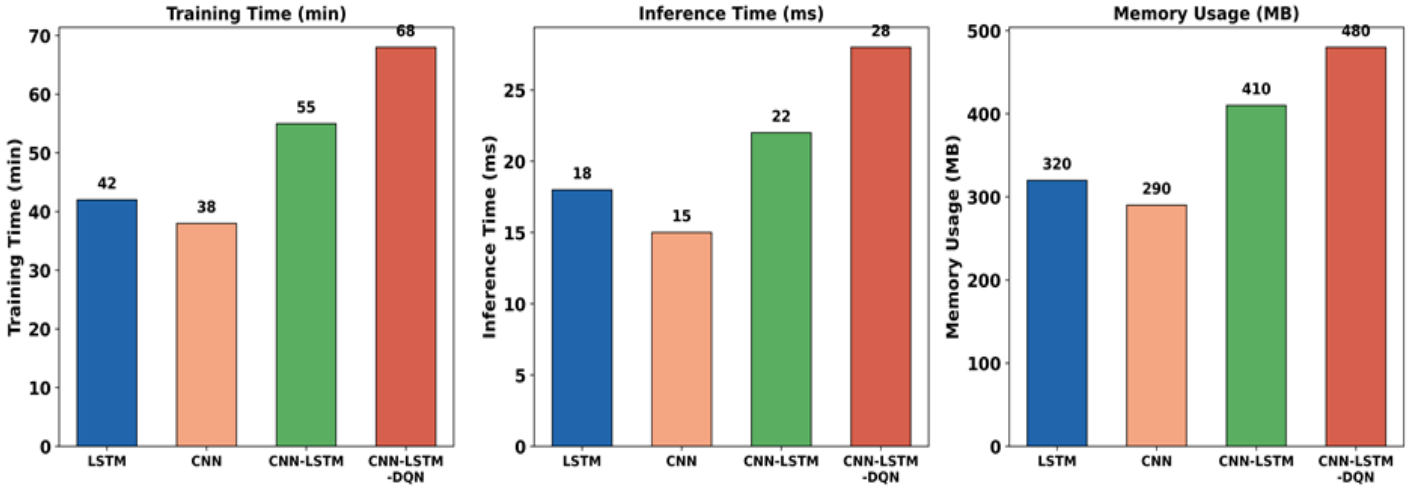

Figure 14. Computational Performance Comparison

Figure 14 compares the cost of computations between models. The suggested CNN-LSTM-DQN needs the most training time (68 min), inference time (28 ms) and memory (480 MB). Although they have become more complex, the benefits of these overheads are a high level of prediction accuracy and traffic optimization performance, which is a good trade-off.

8. Conclusion

In this paper, a new Hybrid CNN-LSTM-DQN Framework in Intelligent Routing and Traffic Prediction in Internet of Vehicles in Dynamic Urban Environments has been provided. The framework suggests a combination of three synergistic deep learning modules, a CNN spatial feature extractor, an LSTM temporal dependency modeller, and a DQN adaptive routing agent, which can be used to face the two-fold challenge of effectively predicting traffic and selecting the most suitable route in dynamic IoV networks. Extensive experiments on three benchmark datasets, including UCI Traffic, VANET Mobility Traces and SUMO Urban Simulation, under three traffic density conditions showed similar and statistically significant performance enhancement across state-of-the-art baselines. The suggested framework demonstrates 22 25 percent improvement of traffic forecasting (RMSE) error, 23 30 percent decrease in average vehicle travel time, 30 40 percent decrease in waiting time and 33 36 percent decrease in the network congestion index. Ablation experiments confirmed that every architectural component adds value to the overall performance, and the CNN-LSTM-DQN complex had the highest results on all metrics considered. All the improvements were found to be at the 0.01 level very significant and thus statistical analysis of significance was conducted. The 28 ms end-to-end inference latency is within operational real time constraints of vehicular routing applications. Although these have been attained, there are limitations to the current study. The main weakness is that, framework validation was only done in the SUMO simulation environment, which, though realistic, does not fully represent the overall complexity, unpredictability and heterogeneity of communication in the real world urban vehicular networks. Also, the existing framework is based on the premise of a trusted V2V and V2I communication which might not necessarily be true in actual deployment conditions.

The future studies will continue to go in a number of complementary ways. First, we will seek to do real-life implementation and testing in live IoV testbeds to determine how it will perform in real-life traffic conditions. Second, it will look into integrating with Vehicle-to-Everything (V2X) communication protocols and 5G network infrastructure. Third, edge AI optimization methods (model pruning, knowledge distillation, and quantization) will be explored to resource-constrained hardware. Fourth, the concept of federated learning will be discussed to tackle the issue of data privacy in multi-stakeholder IoV. Lastly, generalization to multi-agent DRL environments, in which many vehicles jointly optimize their routing policies, is an attractive direction to network-level emergent traffic optimization.

References

1. O. Ekpo, V. Casola, A. De Benedictis, P. Asuquo, and B. Agbor, "A Hybrid CNN–LSTM–Attention Framework for Intrusion Detection in Smart Mobility Networks," Future Internet, vol. 18, no. 4, p. 210, 2026.
2. S. Bhardwaj and R. Saha, "A secure and scalable traffic management framework using hybrid DNN-CNN and geographic routing in V2X networks," Peer-to-Peer Netw. Appl., vol. 19, p. 48, 2026.
3. K. A. Almejalli, "Optimal deep neural network based road traffic management system for Internet of Things based smart city environment," Sci. Rep., vol. 16, p. 12136, 2026.
4. I. Topilin, J. Jiang, A. Feofilova, and N. Beskopylny, "Traffic Flow Prediction via a Hybrid CPO-CNN-LSTM-Attention Architecture," Smart Cities, vol. 8, no. 5, p. 148, 2025.
5. L. Krishnasamy et al., "Intelligent traffic congestion forecasting using BiLSTM and adaptive secretary bird optimizer for sustainable urban transportation," Sci. Rep., vol. 15, p. 18423, 2025.
6. A. Durairaj et al., "An Optimized Deep Learning-Based Smart Parking Mechanism for Smart City Environment," Int. J. Comput. Intell. Syst., vol. 18, p. 290, 2025.
7. R. Tirumalasetti, S. K. Singh, P. K. Roy, and S. Mishra, "A Systematic Review of VANET Routing Protocols for Intelligent Transport Systems," J. Adv. Transp., p. 7999623, 2026.
8. R. Jadhav, A. Hingrajiya, S. Gore, and V. Jain, "Predictive Urban Navigation Using Hybrid TCN–STGCN Traffic Forecasting and Reinforcement Learning," Int. J. Innov. Res. Technol., vol. 12, no. 11, pp. 9412–9415, 2026.
9. S. S. S. Ramesh, J. F. Banu, V. R. Kavitha, and T. Ramesh, "Enhancing Intelligent Transportation Systems in Smart Cities Using VANETs With Deep Reinforcement Transfer Learning and Explainable AI," Trans. Emerg. Telecommun. Technol., vol. 36, no. 8, p. e70219, 2025.
10. J. J. S. et al., "Intelligent traffic prediction system using hybrid convolutional neural networks for smart cities," Multimed. Tools Appl., vol. 84, pp. 31919–31937, 2024.
11. J. Zhang, J. Sha, C. Zhang, and Y. Zhang, "A CNN-LSTM-GRU Hybrid Model for Spatiotemporal Highway Traffic Flow Prediction," Systems, vol. 13, no. 9, p. 765, 2025.

12. M. S. Almutairi, "Deep Learning-Based Solutions for 5G Network and 5G-Enabled Internet of Vehicles," Math. Probl. Eng., vol. 2022, p. 6855435, 2022.
13. N. P. Farazi, B. Zou, T. Ahamed, and L. Barua, "Deep reinforcement learning in transportation research: A review," Transp. Res. Interdiscip. Perspect., vol. 11, p. 100425, 2021.
14. S. Dikshit et al., "The Use of Artificial Intelligence to Optimize the Routing of Vehicles and Reduce Traffic Congestion in Urban Areas," EAI Endorsed Trans. Energy Web, vol. 10, 2023.
15. A. A. Ouallane, A. Bakali, A. Bahnasse, S. Broumi, and M. Talea, "Fusion of engineering insights and emerging trends: Intelligent urban traffic management system," Inf. Fusion, vol. 88, pp. 218–248, 2022.
16. F. Duan, B. Han, and X. Bu, "Synergistic integration of refined pelican optimization algorithm and deep neural networks for autonomous vehicle control in edge computing architectures," Sci. Rep., vol. 15, p. 19338, 2025.
17. Y. Zheng, "Traffic Flow Prediction Algorithm in the Intelligent Internet of Vehicles," in Proc. 6th Int. Conf. Comput. Sci. Manage. Technol., pp. 508–512, ACM, 2026.
18. F. A. Albalooshi, "Advancing Urban Planning with Deep Learning: Intelligent Traffic Flow Prediction and Optimization for Smart Cities," Future Transp., vol. 5, no. 4, p. 133, 2025.
19. S. Buchade and G. Sakarkar, "Exploring and Evaluating Deep Learning Techniques for Traffic Prediction in Urban Environments," Transp. Res. Rec., 2025.
20. S. S. Channamallu et al., "Enhancing Urban Parking Efficiency Through Machine Learning Model Integration," IEEE Access, vol. 12, pp. 81338–81347, 2024.
21. V. N. Skoropad et al., "Dynamic Traffic Flow Optimization Using Reinforcement Learning and Predictive Analytics," Sustainability, vol. 17, no. 8, p. 3383, 2025.
22. W. Chango et al., "Predicting Urban Traffic Congestion with VANET Data," Computation, vol. 13, no. 4, p. 92, 2025.
23. V. Shepelev et al., "Predicting Urban Traffic Congestion Through Deterministic and Stochastic Modeling Using LSTM Neural Networks," Sustainability, vol. 17, no. 23, p. 10655, 2025.
24. Liang Zhao, Olga Gkountouna, and Dieter Pfoser. 2019. Spatial Auto-regressive Dependency Interpretable Learning Based on Spatial Topological Constraints. ACM Trans. Spatial Algorithms Syst. 5, 3, Article 19 (August 2019), 28 pages. DOI:https://doi.org/10.1145/3339823
25. Dataset used Kaggle, “VANET Traffic Congestion Dataset,” 2022. [Online]. Available: https://www.kaggle.com/datasets/ucimachinelearning/vanet-traffic-congestion-dataset
26. Simulation of Urban MObility (SUMO), “Simulation of Urban Mobility (SUMO),” Eclipse Foundation, 2024. [Online]. Available: https://www.eclipse.org/sumo/