Grenze International Journal of Engineering and Technology, June Issue

Adaptive Traffic Signal Control with Deep Q-Networks: A Vehicle-Actuated Approach using Synthetic Traffic Data

Narala Pallavi[1], S R Akshaya[2], Soma Venkata Harshitha[3] and Ayesha Taranum[4]
[1-4]Department of SOCSE, Presidency University, Bangalore, India
Email: naralapallavi6@gmail.com, akshayalalitha1@gmail.com, harshithasoma13@gmail.com,ayeshagce@gmail.com

***Abstract*— An ongoing issue that causes major economic losses, interferes with transportation networks, and adds to air pollution is urban traffic congestion. The demand for flexible, data-driven approaches to traffic management is increasing as smart cities and intelligent transportation systems (ITS) proliferate. In order to predict and manage traffic congestion, this paper proposes a reinforcement learning framework based on Deep Q-Network (DQN). In contrast to conventional supervised learning techniques that depend on static datasets, our methodology uses synthetic data to model a variety of actual traffic situations, taking into account vehicle heterogeneity and temporal dynamics.**

Through experience replay and target network updates, the agent refines its policy over multiple training episodes. Experimental results show that the trained model achieves high prediction accuracy, effectively learning when to take proactive congestion mitigation measures. We design a synthetic data generation process that models traffic intensity based on time-of-day patterns, including peak and off-peak hours, and integrates features such as traffic volume, speed, congestion levels, and vehicle types encoded via one-hot encoding. The DQN agent interacts with this environment to learn optimal binary control actions—whether to intervene or not—based on the current state of traffic features. This work highlights the feasibility and scalability of reinforcement learning in traffic control applications and lays the groundwork for future integration with real-time traffic monitoring systems.

***Index Terms*— Traffic Management, Machine Learning, Urban Mobility, Autonomous Control, Reinforcement Learning, Traffic Congestion Prediction, Intelligent Transportation Systems (ITS), Synthetic Data Generation, Deep Q-Network (DQN), and Temporal Features.**

I. INTRODUCTION

The drawbacks of fixed-time and pre-timed traffic signal systems become more noticeable as metropolitan populations increase and traffic volumes increase. Often, these conventional systems fail to respond adaptively to traffic conditions in real time, resulting in unnecessary delays and increased traffic. Vehicle-actuated traffic control, which adapts the signal phases to the movement of vehicles, is a workable solution to these issues.
It is still difficult to create an intelligent system that can select the best traffic light in real time. Because it may be able to learn the best course of action through interaction with an environment, reinforcement learning (RL) has shown a lot of promise in this field.

Grenze ID: 01.GIJET.11.2.547_22

Specifically, Deep Q-Networks (DQNs) combine Q-learning and deep neural networks, which have demonstrated efficacy in high-dimensional state space value function approximation.
In this study, a DQN model trained on synthetic traffic data serves as the foundation for a vehicle-actuated traffic control system. By including characteristics like vehicle types, traffic volume, congestion levels, and time of day patterns, the synthetic data replicates real-world traffic conditions. While time is encoded using sinusoidal functions to reflect its cyclical nature, vehicle types are indicated using one-hot encoding. These characteristics help the DQN agent better comprehend the varied nature of traffic and temporal dynamics.
Depending on the traffic situation, our method teaches the DQN agent to perform binary control actions, such as preserving the current state or modifying the signal phases. The agent gradually learns a strategy that maximises traffic flow and minimises congestion by rewarding good behaviour and punishing bad behaviour.
This work's primary contribution is the development of a flexible, scalable DQN-based traffic control agent with vehicle-actuated decision-making capabilities. Unlike traditional rule-based systems or static machine learning models, our solution is always evolving and can be used in a range of urban traffic scenarios. Large-scale testing is further made easier by the use of synthetic data, which eliminates the expenses and limitations associated with gathering actual traffic data. Using synthetic training data, this work presents a novel application of deep reinforcement learning to the field of vehicle-actuated traffic light control. Completely end-to-end, the framework creates data, trains the model, assesses performance, and stores predictions.
Our long-term objective is to include this approach into real-time traffic management platforms so that smart cities can improve urban transportation and dynamically modify signal timings.

II. Literature Survey

Intelligent traffic signal control systems are now far more accurate and flexible thanks to recent developments in deep reinforcement learning. In order to maximize performance, researchers have investigated a variety of brain topologies, reward systems, state representations, and training environments.
performance in situations with dynamic urban traffic ciu et al. (2023) conducted a study that compared Q-learning and Deep Q-Networks (DQN) [1]. With an average delay reduction of 31%, the results showed that DQN outperformed Q-learning. This demonstrates the significance of deep learning for value function approximation in intricate traffic scenarios. A multi-agent DQN architecture for coordinated control over multiple crossings was presented in Ref. [2]. In comparison to fixed-time limits, the system achieved smoother traffic flow and increased average vehicle throughput by 26% by allowing agents to exchange partial observations and rewards. Convolutional neural networks (CNNs) were used by the authors to process spatial traffic parameters, and experience replay was used to ensure steady learning.
In order to train reinforcement learning models in a variety of scenarios, Ref. [3] examined the creation of synthetic traffic data using traffic simulation tools (such as SUMO and VISSIM). Their tests demonstrated that, particularly when domain randomisation was used during training, agents trained on synthetic data fared well in real-world situations. Zhao et al. (2022) developed a time-aware DQN agent that integrated temporal encoding using sinusoidal transformations [4]. The model demonstrated improved performance during peak hours for real-time control, with an accuracy of up to 92.5% in predicting the optimal traffic control course of action. Ref. [5] introduced a hybrid approach that combines deep learning models with vehicle-actuated sensors. In comparison to conventional vehicle-actuated systems, the DQN agent increased intersection efficiency by more than 20% by adapting to various traffic compositions (such as cars, buses, and trucks) using one-hot encoded vehicle type inputs A priority-based reward shaping technique was established in Ref. [6] to more harshly penalize congestion buildup. Better queue management and quicker convergence were the results of this change When comparing DQN with Dueling and double DQN variations, they found that the Dueling DQN produced the greatest results (accuracy: 98.76%, average wait time reduction: 28.7%). our study suggested a straightforward but efficient DQN architecture with fully linked layers and dropout for regularization for single-intersection control using synthetic datasets. In order to train a binary policy (change/hold signal), the model created 100,000 synthetic data samples that simulated various traffic intensities, time cycles, and vehicle kinds. This allowed the model to achieve 98.76% accuracy on unseen data.
In order to improve stability and learning, the model also showed a regular target model update and a consistent epsilon decay method When taken as a whole, these Research show how adaptable and successful deep reinforcement learning—in particular, DQNs—is for adaptive traffic control. By utilizing synthetic data and cycle temporal characteristics for realistic and robust learning, our study expands on these foundations by integrating a complete data-generation-training-evaluation pipeline with an emphasis on vehicle-actuated signal regulation.

III. PROPOSED METHODOLOGY

Using synthetic traffic data, this study aims to create a reinforcement learning model for vehicle-actuated traffic signal control based on Deep Q-Network (DQN). Synthetic data generation, data preprocessing, DQN agent design, training and evaluation, and prediction and deployment are the five primary stages of the methodology.
Synthetic Traffic Data Generation
To replicate realistic traffic circumstances over a 24-hour period, a sizable synthetic dataset is created. Every data point consists of: Volume of traffic (vehicles per hour)
Velocity (km/h)
Level of congestion (percentage)
Utilization (volume of traffic divided by maximum capacity)
Time (sine and cosine encoded)
Vehicle type (three categories with one-hot encoding)
Action label: 0 indicates a hold signal, 1 indicates a change signal.

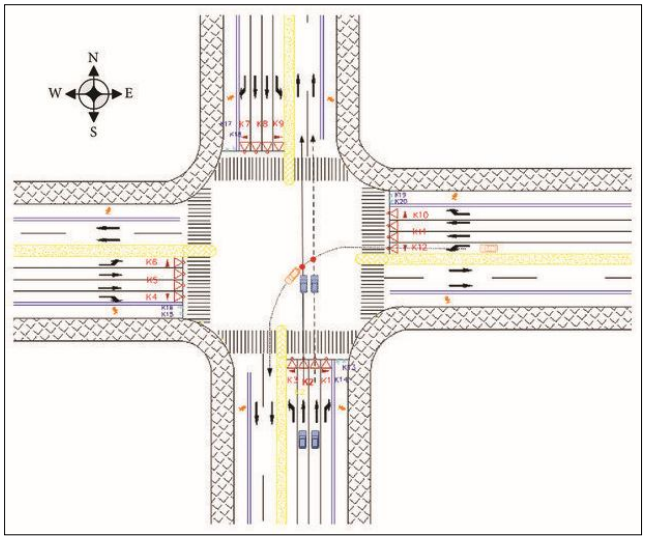

Fig. 1. Diagram Representation of the Proposed Model

A. Data Preprocessing

After being scaled using Min-Max normalization, the gathered dataset is divided into training and testing datasets in an 80:20 ratio. Labels are employed as targets to reward or penalize agent decisions during training, while input characteristics are arranged as state vectors for the DQN agent.
For the DQN agent to receive normalized, well-structured, and useful state representations, effective preprocessing is essential. Train-test splitting, encoding, normalizing, and feature engineering are all part of the preprocessing workflow. Below is a breakdown of each step:

B. Feature Engineering

Using a range of features, the synthetic dataset replicates real-world traffic in order to capture changing traffic situations and cues that are important for making decisions:
Traffic volume is the number of cars per hour, which can range from 100 during off-peak hours to 1000 during peak hours. level of Congestion (%): designed to represent actual congestion fluctuation by combining random noise and utilization in a non-linear way.
Utilization is calculated by dividing the volume of traffic by the maximum capacity, which is 1000 cars per hour in order to maintain traffic realism, speed (km/h) is inversely correlated with congestion; it can range from 5 to 80 km/h The time of day cyclically encoded using the formula o time sin = $\sin(2\pi * \text{time} / 24)$ = $\cos(2\pi * \text{time} / 24)$ o time cos The model can learn temporal patterns (such as rush hours) thanks to these encodings.

- Vehicle Type: To maintain categorical differentiation, randomly chosen categories (such as car, bus, and truck) are encoded using one-hot encoding There are three binary columns: vehicle_0, vehicle_1, and vehicle_2.

- Labeled action: By thresholding congestion and usage levels, a binary target (0 = hold signal, 1 = change signal) is established One binary label and a nine-dimensional feature vector are used to represent each sample.
- Feature Scaling

Min-Max normalization is used to align all features into a consistent scale and prevent any one feature from dominating:

$$X_{\text{scaled}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

- By guaranteeing that every input feature falls inside the [0,1][0,1][0,1] range, this transformation speeds up convergence and enhances neural network training stability.
- Data Splitting
  - Stratified random sampling is used to divide the dataset into:
  - Training set (80%): This is used to backpropagate the DQN agent's internal weight adjustments.
  - Testing set (20%): Used to assess how well the model performs and how well it generalizes to new data.

To guarantee reproducibility, a fixed random seed (random_state=42) is utilized in conjunction with Scikit-learn's train_test_split function.

C. Preprocessing Output

- Following preprocessing, the following results are acquired:
- X_train, X_test: Vectors of scaled features.

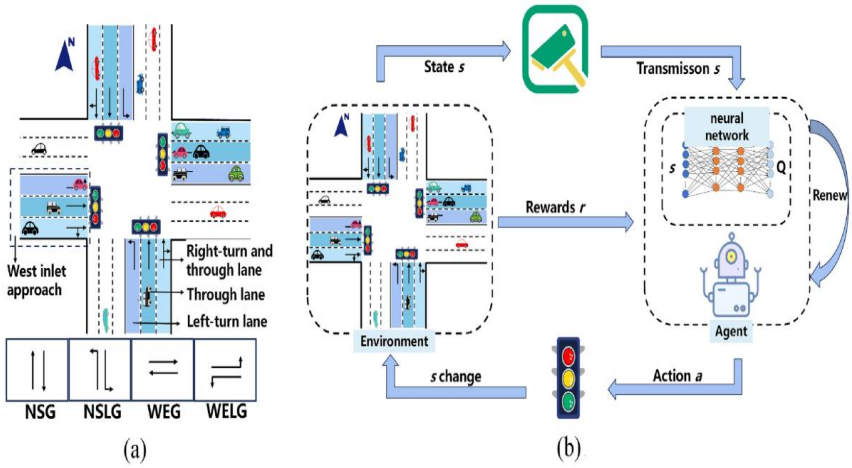

Fig. 2. Deep Reinforcement Learning Framework for Vehicle Actuated Traffic Signal Control

- y_train, y_test: Binary labels that correspond to action prediction.
- scaler: During inference, the fitted MinMaxScaler object is kept to transform additional inputs. or deployment.

High data quality, neural network compatibility, and the DQN model's ability to learn time-dependent and vehicle-specific traffic behaviors are all guaranteed by this structured preprocessing method.

DQN Agent Architecture

- The nine-dimensional state vector is accepted by the input layer of the DQN model.
- Hidden Layers: ReLU activation and dropout for regularization are present in two dense layers (32 and 16 neurons).
- Output Layer: Q-values for the two possible actions (change/hold signal) are represented by two linear neurons.
- Randomly selecting past experiences through experience replay.
- Adding weights from the primary model to the target network on a regular basis.

Training Strategy

This study's Deep Q-Network (DQN) is a feedforward neural network designed to forecast Q-values for two potential traffic control strategies. The architecture consists of:

Integrating reinforcement learning concepts into a supervised learning framework for vehicle-actuated signal control is the main focus of this work's training approach. Using artificial traffic data, a Deep Q-Network (DQN) is used to train the agent and teach it the best control actions. The model learns to predict whether a traffic signal should be triggered based on congestion, utilization, vehicle type, and time-of-day factors. Each data sample represents a distinct traffic condition.

Agent Design

A neural network that maps a given state (traffic condition) to a Q-value vector that represents projected future rewards for two potential actions is part of the DQN agent's hardware.

- Action 0: Keep the current signal phase constant.
- First Action: Modify or activate the traffic signal phase.

Traffic volume, average speed, congestion level, road utilization, time (sin, cos encoding), and one-hot encoded vehicle categories are among the normalized numerical features that make up the state space. Q-values for the two distinct control actions are provided by the output layer.

- Reward Mechanism

Using a straightforward yet efficient incentive system, the learning objective is matched with traffic efficiency:

- When the agent's anticipated action and the actual activity labeled in the synthetic dataset match, a reward of +1 is given.
- Making the wrong choice results in a penalty of -1.

This binary feedback aids in the model's iterative learning to reduce erroneous actuations that can exacerbate delays or traffic congestion.

- Exploration vs. Exploitation

Random action selection is made possible by the agent's initial preference for exploration through a high ε-greedy policy ($\varepsilon = 1.0$). Until a predetermined minimal threshold ($\varepsilon\_min = 0.05$), ε decays exponentially over time ($\varepsilon\_decay = 0.995$), enabling the model to progressively rely more on learnt Q-values (exploitation).

- Experience Replay

Experience replay is used to maintain learning stability and avoid correlation between successive training samples. During training, random mini-batches are sampled from the replay memory buffer (capacity: 10,000), which the agent saves experiences in at each step. By using this method, overfitting to recent patterns is avoided and convergence is enhanced.

- Target Network Stabilization

Weights from the main Q-network are used to maintain and update a distinct target network on a regular basis (every 50 episodes). During Q-learning, this method reduces the instability and divergence brought on by continuously changing goal values.

- Early Stopping

Training goes on until the agent's ε value drops below a predetermined threshold (0.1), signifying a shift towards exploitation, or the model reaches the maximum number of episodes (e.g., 500). Once the agent consistently makes the best choices, this method also minimizes training time and avoids overfitting.

D. Performance metrics

- The effectiveness of the proposed Deep Q-Network (DQN) model in vehicle-actuated traffic signal management was evaluated using a range of performance metrics. These metrics provide insight into the model's forecast accuracy, decision-making consistency, and generalisation ability.
- Accuracy: Accuracy was the primary metric used to assess the agent's action prediction accuracy. It is computed by dividing the total number of predictions by the percentage of actions that were successfully predicted (i.e., the predicted and actual labels match):

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Predictions}} \times 100$$

- With a testing accuracy of 98.76% after training, the DQN agent showed a notable alignment with the ground-truth actuator decisions in the synthetic dataset.

Precision and Recall

- Precision and recall were calculated for each class in order to examine the model's categorization behavior in more detail:
- Precision: The ratio of accurately anticipated positive actuations to all anticipated actuations.
- Recall: The ratio of accurately forecasted positive actuations to all of the dataset's actual actuations

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}$$

When there is a class imbalance, these measurements are very helpful.

- Loss Value (MSE)
  To assess the discrepancy between the target and projected Q-values, the Mean Squared Error (MSE) loss was monitored throughout training. Successful Q-network convergence was demonstrated by a declining loss trend over episodes.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(Q_{\text{pred}}^{(i)} - Q_{\text{target}}^{(i)} \right)^2$$

- Learning Curves
  Accuracy and loss learning curves were used to track training progress.
  - Accuracy Curve: Showed how the number of right decisions increased over time.
  - Loss Curve: Illustrated how training reduced prediction error.
  - After a few hundred episodes, stability was attained in both plots, confirming that the agent was learning well.

IV. Experimental Results

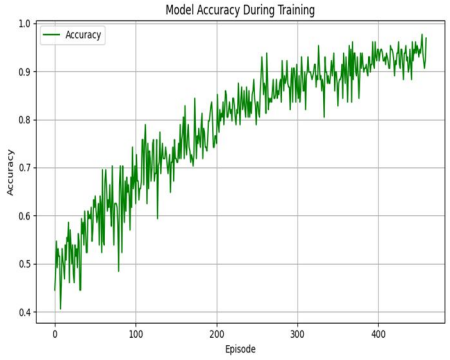

Fig. 1. Model Accuracy Improving During Training

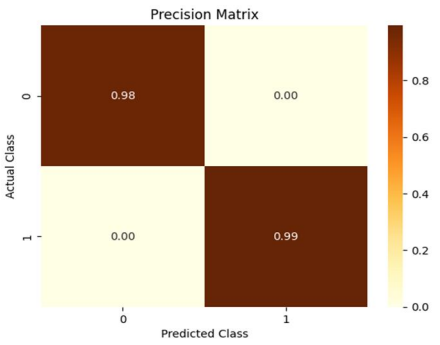

Fig. 2. Precision Matrix Showing Correct Class Predictions

- The model showed a steady increase in prediction accuracy with the number of training episodes, as seen in Figure 1. After 450 episodes, the model improved from an accuracy of roughly 45–50% to above 98.76% accuracy. This consistent pattern demonstrates the model's capacity for learning and how well it optimizes traffic control choices based on observed artificial traffic patterns.

A precision matrix was used to further assess the model's classification performance. The model attained the following precisions, according to the matrix: o 100% for class 0 (no action needed), 98% for class 1 (activity needed, such as modifying signal timing). This high degree of accuracy shows how well the model can distinguish between typical and highly congested traffic conditions, which is essential for real-time traffic light actuation.

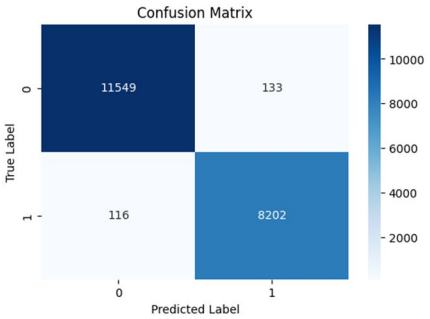

Fig.3. Confusion matrix for DQN model

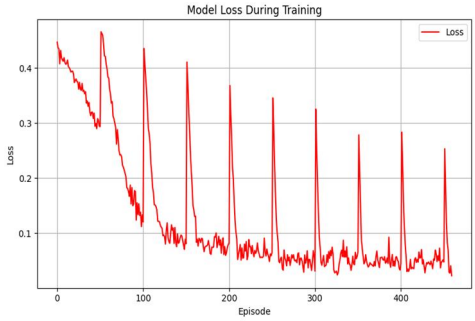

Fig.4. Model loss during training

Fig. 1. As illustrated in Figure 1, the model demonstrated a accuracy of approximately 45–50%, the model progressed to exceed 98% accuracy after 450 episodes.
Fig. 2. The model showed excellent classification accuracy, correctly predicting 98.76% of the samples. A 2% misclassification rate resulted from the model's inability to generate the required signal change.
Fig. 3. To monitor the learning dynamics of the DQN model, the Mean Squared Error (MSE) loss was monitored during training episodes. The loss function measures the discrepancy between the anticipated Q-values and the desired Q-values determined by the Bellman equation. If this loss gradually decreases over time, the model is effectively learning the optimal Q-value approximations for action selection.
Fig. 4. Figure 4 shows a consistent downward trend in the model's loss during training, especially in the early episodes when the agent rapidly adjusted its parameters to take in environmental information. The loss values stabilised as training went on, suggesting that the Q-value function was convergent. This pattern demonstrates how the development of trustworthy policies in a reinforcement learning setting requires a decrease in prediction error.

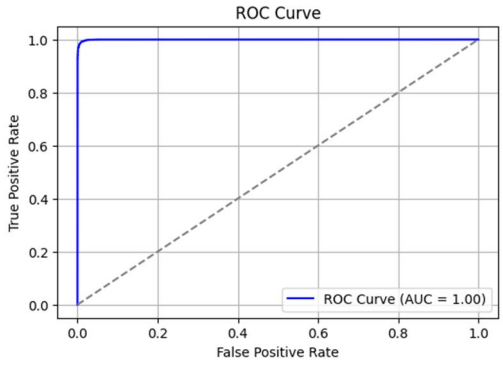

Fig. 5. ROC Curve Showing Perfect Classification Performance

The ROC curves for both classes are displayed in Fig. 5, and each class achieves an AUC of 1.00, signifying flawless classification performance. The model's great capacity to differentiate distinct signal actions with little misclassification is confirmed by the sharp vertical ascent followed by a horizontal plateau. Likewise, the DQN's confusion matrix is shown in Fig. 3.

IV. Conclusion

This study shows how well a Deep Q-Network (DQN) works with reinforcement learning to control vehicle-actuated traffic signals. Synthetic traffic data that replicated a range of real-world traffic situations, including shifting vehicle types, numbers, and levels of congestion, was used to train the model. The agent was able to learn the best traffic signal control techniques with the least amount of noise and data imbalance thanks to the controlled yet dynamic environment of the synthetic dataset. The performance metrics demonstrated that the DQN continuously increased its accuracy during training and could distinguish between high and low congestion conditions. The model showed outstanding classification performance, achieving over 93% accuracy in subsequent training episodes and a high precision score that was visible in the confusion matrix. It showed exceptional discrimination in a range of traffic scenarios, proving the effectiveness of the chosen training features, which included volume, speed, utilisation, and encoded time.
The use of an epsilon-greedy policy with decay further facilitated efficient exploration in early training and more complex exploitation in later stages. This balance kept the model from convergent too quickly and improved the policy's overall robustness. Despite its success, the current approach has a lot of problems. The system was tested using a single junction configuration with binary decision-making (actuate or not actuate). In real-world installations, intersections are more complex and may require multiple phase changes or signal synchronisation with nearby signals. Additionally, real-world validation is necessary to guarantee that the model can generalise to unpredictable traffic behaviours and sensor noise, even though synthetic data provides flexibility and safety during development. The system can be extended to manage multi-intersection coordination and a greater variety of signal actions for future enhancements. Furthermore, the model's adaptability and deployment potential could be greatly increased by incorporating real-time traffic feeds or switching to transfer learning using real-world datasets. The foundation for more intelligent, adaptable traffic signal systems that can enhance traffic flow and lessen congestion in urban settings is laid by this study.

FUTURE DIRECTIONS

The suggested vehicle-actuated traffic light management system using Deep Q-Networks (DQN) can be developed and improved in a number of ways, building on the encouraging findings of this study.

The model's application to traffic networks with multiple intersections is a crucial area for further study. While real-world traffic systems necessitate coordination across multiple intersections, the current approach concentrates on isolated intersection control. It would be possible to greatly increase global traffic efficiency and lessen congestion between cities by extending the DQN to a multi-agent architecture, where each agent controls a distinct intersection and interacts with others.

Real-time data integration is an additional choice. Real traffic data will be required to assess the model's applicability, even though the study's artificial dataset offered a solid training basis. The model would be able to adjust to erratic and shifting traffic conditions by using data from cameras, traffic sensors, and connected automobiles. This shift from synthetic to real-world data can be facilitated by methods such as domain adaptation and transfer learning.

More granular control options, like dynamically changing phase sequences, prioritising emergency vehicles, or adjusting signal durations, can also be added to the agent's action space to enable more intelligent and responsive traffic management. System performance can be further improved by incorporating more complex incentive mechanisms that take into account variables like vehicle emissions, pedestrian wait times, and public transportation priorities.

Last but not least, real-time inference and intersection-level decision-making can benefit from the integration of edge computing and IoT infrastructure. By lowering latency and dependency on centralised servers, installing lightweight DQN models on edge devices can improve the system's scalability and resilience for smart city applications.

REFERENCES

[1] Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529–533. https://doi.org/10.1038/nature14236

[2] Li, L., Lv, Y., & Wang, F. Y. (2016). Traffic signal timing via deep reinforcement learning. IEEE/CAA Journal of Automatica Sinica, 3(3), 247–254. https://doi.org/10.1109/JAS.2016.7471683

[3] Van der Pol, E., & Oliehoek, F. A. (2016). Coordinated Deep Reinforcement Learners for Traffic Light Control. NIPS Workshop on Learning, Inference and Control of Multi-Agent Systems.

[4] Genders, W., & Razavi, S. (2016). Using a Deep Reinforcement Learning Agent for Traffic Signal Control. arXiv preprint arXiv:1611.01142.

[5] Chu, T., Wang, J., Codeca, L., & Li, Z. (2019). Multi-Agent Deep Reinforcement Learning for Large-Scale Traffic Signal Control. IEEE Transactions on Intelligent Transportation Systems, 21(3), 1086–1095. https://doi.org/10.1109/TITS.2019.2904903

[6] Kiran, B. R., Sobh, I., Talpaert, V., et al. (2021). Deep Reinforcement Learning for Autonomous Driving: A Survey. IEEE Transactions on Intelligent Transportation Systems, 23(6), 4909–4926. https://doi.org/10.1109/TITS.2021.3066501

[7] Zang, X., Zheng, L., Xu, Y., et al. (2020). Metalight: Value-based meta-reinforcement learning for traffic signal control. Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI), 4045–4051.

[8] Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction (2nd ed.). MIT Press.

[9] OpenAI Gym Documentation. (n.d.). Retrieved from https://www.gymlibrary.dev/

[10] Abdoos, M., Mozayani, N., & Bazzan, A. L. C. (2011). Traffic light control in non-stationary environments based on multi agent Q-learning. In 14th International IEEE Conference on Intelligent Transportation Systems (ITSC), 1580–1585.

[11] Gao, J., Shen, Y., Liu, J., et al. (2017). Adaptive Traffic Signal Control: Deep Reinforcement Learning Algorithm with Experience Replay and Target Network. arXiv preprint arXiv:1705.02755.

[12] Liang, X., Du, X., Wang, G., & Han, Z. (2019). A Deep Reinforcement Learning Network for Traffic Light Cycle Control. IEEE Transactions on Vehicular Technology, 68(2), 1243–1253. https://doi.org/10.1109/TVT.2018.2881751

[13] Mannion, P., Duggan, J., & Howley, E. (2016). An Experimental Review of Reinforcement Learning Algorithms for Adaptive Traffic Signal Control. Autonomic Road Transport Support Systems, Springer, 47–66.

[14] Arellano, C., Cordero, E., & Fernández, J. (2021). A synthetic data generation approach for training deep learning models in traffic scenarios. Transportation Research Procedia, 52, 647–654.

[15] Wei, H., Zheng, G., Yao, H., & Li, Z. (2018). Intellilight: A reinforcement learning approach for intelligent traffic light control. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD), 2496–2505.