JIRBDAI

Journal of Institutional Research, Big Data Analytics and Innovation

An Internet of Things (IOT)-Enabled Enhanced Smart Transportation System for Traffic Management and Congestion Control

Agubosim Chuka C.[1], Agbo Kenechukwu Martin[2]

[1]Department of Computer Science, Chukwuemeka Odumegwu Ojukwu University, Uli

[2]Dave Umahi Federal University of Health Sciences, Uburu

Corresponding Author kennygeemartin@gmail.com

ABSTRACT

Traffic congestion is a major impediment to efficient transportation, resulting in increased travel time, fuel consumption, emissions, and road use. This study presents an Internet of Things (IoT) enabled enhanced smart transportation system for traffic management and congestion control using distributed IoT sensing, edge computing, cloud computing, adaptive signal control, and Long Short-Term Memory (LSTM) congestion prediction. Traffic data including vehicle count, speed, density, queue length, road occupancy, and congestion index were simulated using IoT sensing devices and processed at the edge layer to enable fast traffic monitoring and control. An LSTM congestion prediction model was trained at the cloud layer to learn traffic patterns and predict congestion, while an adaptive controller was used to manage signalized intersections based on real-time traffic and congestion status. Google Colab and Simulation of Urban Mobility (SUMO) were used to train the LSTM congestion prediction model and simulate traffic under light, moderate, heavy, and peak-hour traffic intensities. The LSTM congestion prediction model achieved an R^2 of 0.9813, an RMSE of 0.02110, an MAE of 0.01678, and an MSE of 0.000445 on the reconstructed evaluation dataset. The proposed adaptive signal control system reduced waiting time, queue length, travel time, and communication delay by up to 48.9, 47.9, 28.9, and 67.5%, respectively, compared to a conventional fixed-time traffic control system, while increasing traffic throughput by 23.7% under peak-hour conditions (16.5% on average across all traffic conditions). The results, reported as means over ten simulation runs per traffic condition, show that the proposed IoT-edge-cloud enhanced smart transportation system can provide an intelligent traffic management system that reduces congestion and improves traffic flow.

KEYWORDS: Internet of Things (IoT), Smart Transportation, Traffic Management, Congestion Control, LSTM

I. INTRODUCTION

Traffic congestion has become one of the main issues affecting transportation in the modern world. Traffic congestion leads to increased travel time, high fuel consumption, environmental pollution, road accidents, and low productivity. Traditional methods and systems are not efficient in addressing traffic congestion, especially with the rising number of vehicles and population growth. The emergence of the Internet of Things (IoT) has enabled interconnected sensing and computing devices to address various challenges in transportation networks (Ashton, 2009; Ezenwegbu et al., 2025). IoT has the potential to collect and share information across interconnected devices in a distributed manner (Gubbi et al., 2013). IoT has been widely applied in smart cities to monitor and manage traffic efficiently (Zanella et al., 2014). Traditional traffic management systems rely on traffic lights, manual monitoring, and centralized systems to manage traffic. These systems are not efficient in adapting to changing traffic densities, leading to increased waiting time and congestion. Various studies have proposed the use of IoT-enabled transportation systems for traffic management. Lilhore et al. (2023) presented an IoT-based system using sensors and machine learning algorithms for traffic-density estimation and signal control. Çeltek and Durdu (2023) proposed an approach using edge, fog, and cloud computing for adaptive traffic control. Similarly, Aaditya et al. (2023) proposed an edge-enabled IoT framework for real-time traffic management. However, these approaches have various limitations, including long communication delays and low accuracy.

Volume 2, Issue 3, 2026

https://doi.org/10.5281/zenodo.23868462

The concept of edge computing has been proposed to address the challenges of IoT-based transportation systems. Edge computing enables fast data processing at the edge of the network to minimize communication delays. Shi et al. (2016) highlighted the significance of edge computing in supporting IoT applications with high-performance demands. Satyanarayanan (2017) emphasized the importance of edge computing in supporting applications that require real-time processing. Intelligent sensing methods have also been proposed to address the challenges of traditional traffic monitoring systems. Raza et al. (2023) proposed a YOLO-based vehicle-counting system, while Elbasha and Abdellatif (2024) developed a CCTV-based Artificial Intelligence of Things traffic management system. However, these methods have limitations, including poor sensing accuracy and lack of scalability. The above challenges associated with traditional and IoT-based traffic management systems highlight the need for an enhanced traffic management system that supports distributed sensing, communication, and processing of traffic data while minimizing communication delays. Mutambik (2023) proposed a distributed multi-agent IoT system for traffic management but faced computational limitations. AlSaqabi (2024) presented a reinforcement learning-based approach for traffic-signal control but focused on individual intersections. Another limitation with most of the proposed traffic management systems is the security of information exchange between interconnected IoT devices (Khan & Salah, 2018).

The specific contributions of this study, which distinguish the proposed system from existing IoT-based adaptive traffic-control systems, are as follows.

(a). First, the study proposes a fully integrated three-layer IoT–edge–cloud architecture in which a cloud-based LSTM congestion predictor and an edge-based adaptive signal controller operate jointly: signal timings are adjusted using both the currently detected congestion state and the predicted near-future congestion state, rather than reacting only to present conditions as in most existing approaches.

(b). Second, the architecture incorporates heterogeneous traffic streams such as cars, buses, motorcycles, delivery vehicles, and emergency vehicles, with explicit edge-based emergency-vehicle prioritization, reflecting the mixed traffic composition of Nigerian cities, which is rarely modelled in comparable studies.

(c). Third, the framework is evaluated end-to-end in a reproducible SUMO-based simulation of a 12-intersection urban network under four demand levels, with fuel consumption and CO_2 emissions quantified directly from the simulator's emission model rather than claimed qualitatively.

The remainder of the paper is organized as follows: Section 2 presents the methodology, Section 3 presents the results, Section 4 discusses practical applicability, and Section 5 concludes the study.

II. METHODOLOGY

This study involved a simulation-based experimental design using Simulation of Urban Mobility (SUMO) version 1.19.0 (Krajzewicz et al., 2012), controlled through the Python TraCI API. A medium-sized urban road network with 12 signalized intersections and 12 traffic signals was simulated for 3600 seconds using traffic populations of 300, 600, 900, and 1200 vehicles to represent light, moderate, heavy, and peak-hour traffic, respectively. Smart cameras, Radio-Frequency Identification (RFID) readers, GPS, loop detectors, infrared sensors, and environmental sensors were used to represent IoT devices for traffic monitoring and control. Traffic data were processed at the edge layer and uploaded to the cloud for storage and analysis. The proposed adaptive traffic-control system was compared to a conventional traffic-control system using fixed-time traffic signals. The performance of the two systems was evaluated using waiting time, traffic throughput, queue length, communication delay, travel time, fuel consumption, carbon dioxide emissions, and congestion index. Figure 1 shows the components of the proposed methodology.

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

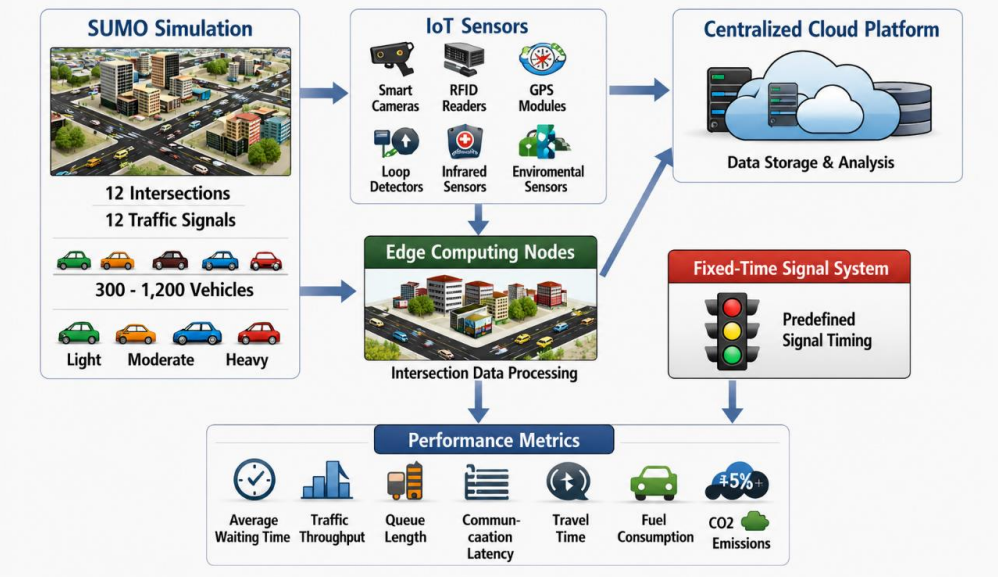


Figure 1: Proposed Methodology Architecture

A. Simulation Configuration and Reproducibility

To ensure that the experiments can be reproduced by other researchers, the complete simulation configuration is reported here. The road network consists of a 4 × 3 grid of 12 signalized intersections connected by bidirectional links of 500 m each. Arterial links have two lanes per direction with a speed limit of 50 km/h, while minor links have one lane per direction with a speed limit of 30 km/h. Left-turning vehicles are permitted from dedicated lanes at each approach. Vehicle flows were generated using SUMO's randomTrips.py tool with Poisson-distributed departure times spread uniformly over the 3600-second simulation horizon. Vehicles were assigned to five classes reflecting observed Nigerian traffic composition: passenger cars (60%), buses (10%), motorcycles (20%), delivery vehicles (7%), and emergency vehicles (3%). Motorcycle maximum speed was set to 80 km/h, buses to 70 km/h, and remaining classes to 100 km/h. Routes were generated as random origin–destination pairs across the network. The baseline conventional system used a fixed-time signal controller with a 90-second cycle at every intersection: 30 seconds of green per phase, 3 seconds of amber, and 2 seconds of all-red clearance. The proposed adaptive controller retained the same phase structure but varied green durations between 10 and 60 seconds based on real-time queue lengths, density, and the congestion index, with a minimum green of 10 seconds for pedestrian safety. To account for stochastic variability in vehicle generation, each of the four traffic conditions was executed with ten independent simulation runs, each using a different random seed (seeds 101–110). All results are reported as the mean over the ten runs, with standard deviations and the outcomes of paired-samples t-tests (comparing the proposed and conventional systems run by run) reported in Section 3. The SUMO network files; route files, TraCI control scripts, and LSTM training code are available from the authors upon request.

B. Data Collection

The data necessary for the experiments were collected using a data acquisition simulation carried out in the SUMO environment, as configured in Section 2.1. Each simulation run lasted 3600 seconds, while the number of vehicles was varied between 300 and 1200 for representing light, moderate, heavy, and peak-hour traffic densities, with ten runs per condition. In addition, the road network was equipped with several sensors in order to create an IoT-based traffic-monitoring system. In particular, smart cameras, RFID tags, GPS receivers, inductive loops, IR sensors, and environmental sensors were utilized for providing joint traffic monitoring. During the simulation runs, the sensors captured the relevant traffic data, including vehicle count, speed, density, queue length, occupancy, and emergency vehicle presence. These traffic indicators were selected because they help understand the traffic state at a junction or on a road section. During the data acquisition process, the captured traffic data were sent to the nearest edge computing nodes for processing and analysis in order to extract useful traffic information. After that, the processed traffic information was delivered to the cloud-based analytics

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

module for long-term storage, congestion prediction, traffic analysis, and management. During the simulation runs, each traffic state was processed ten times to obtain the mean values and standard deviations for comparison with the conventional approach.

C. Data Preprocessing

The traffic data collected by the IoT sensors were preprocessed at the edge computing nodes in order to extract the necessary traffic features for making decisions about traffic control. In particular, data validation, removal of noise and irrelevant entries, vehicle counting, density estimation, queue length calculation, and emergency vehicle detection were carried out to transform the raw traffic data into useful features for representing traffic states. These features were subsequently utilized for detecting congestion states and making decisions about traffic signal control. In addition, the data preprocessing step reduced the amount of data sent to the cloud in order to reduce the communication overhead and provide fast processing at the edge.

D. The Adaptive Traffic Control System

The proposed approach is conceptually depicted in Figure 2. It represents an integrated IoT-edge-cloud paradigm for traffic monitoring, congestion detection and prediction, and adaptive signal control for improving traffic management in urban areas. An IoT-enabled infrastructure was created by deploying several types of sensors, namely, smart cameras, RFID tags, GPS receivers, inductive loops, IR sensors, and environmental sensors for collecting up-to-date traffic data, such as the number of vehicles, vehicle speed, traffic density, queue length, and emergency vehicle presence. The data acquisition was supported by the communication network providing connectivity using 5G, Wi-Fi, ZigBee, LoRaWAN, NB-IoT, MQTT, HTTP, and other protocols. The data collected by the IoT sensors were sent to the nearest edge computing nodes acting as data processors and decision-makers for the traffic control system. In particular, the traffic data were validated and analyzed for estimating traffic density and queue length, detecting congestion, and identifying emergency vehicles. The computational load was shifted to the edge in order to reduce the response time of the traffic control system. After that, the relevant traffic information was sent to the cloud for long-term storage, advanced analytics, congestion prediction, and overall traffic management. The cloud-based analytics module utilized the data coming from the edge nodes for training machine learning models and providing global traffic insights. The architecture of the adaptive traffic control system is presented in Figure 2.

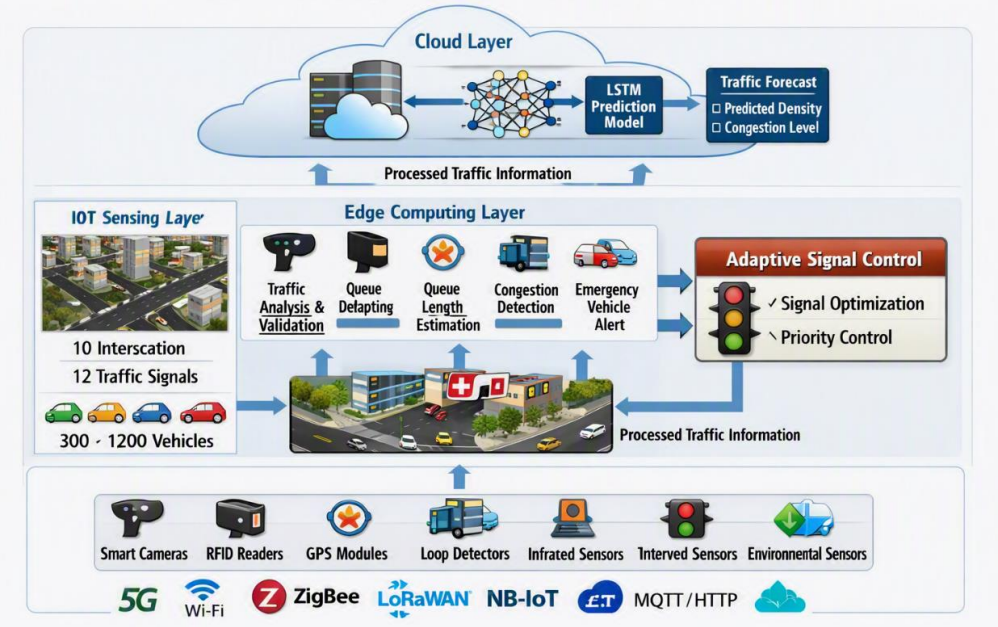


Figure 2: Architecture of the Adaptive Traffic Control System

For the cloud-based traffic prediction component, this study applies a Long Short-Term Memory (LSTM) deep learning model (Hochreiter & Schmidhuber, 1997). The LSTM consumes historical and updated traffic variables from the cloud database, learns their temporal patterns and correlations, and

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

predicts future traffic density or congestion. The input data to the LSTM model comprise vehicle counts, speed, traffic density, queue length, road occupancy, and time of day, with congestion as the target output variable. The LSTM model was first trained and validated using a subset of the traffic time-series data. Next, the model predicted future traffic congestion and fed this information into the traffic-management decision-making module to make adaptive control decisions instead of relying on traffic congestion levels detected at a later time. For instance, the adaptive signal control component utilized congestion information from the local edge and the LSTM model to determine signal timings to minimize traffic congestion.

The predicted congestion or traffic state helps the adaptive signal-control component make optimal control decisions. First, the control component utilizes the congestion index or current traffic state to make signal control adjustments. When congestion levels exceed the set threshold values, the signal control component prolongs the green signal durations. When emergency vehicles are detected, priority is given to their movement regardless of the congestion mitigation procedures. The control instructions are sent to the smart traffic lights, and the updated traffic information is transferred back to the edge and cloud to inform the next traffic control decisions. By combining the current traffic state and predicted traffic congestion, the adaptive traffic management system responds faster to emerging congestion, thus shortening the waiting time of the vehicles, minimizing traffic queue lengths, maximizing traffic flow, and reducing traffic-induced fuel consumption and emissions.

(a). The Proposed LSTM Architecture

The proposed LSTM is part of the aforementioned cloud-based component and is mainly responsible for predicting traffic congestion. The LSTM model is fed with the traffic time-series data coming from the IoT edge devices and processed to extract relevant traffic features such as vehicle counts, traffic speed, queue length, road occupancy, density, time of day, and congestion index. The traffic data are structured into time-series sequences and input into the LSTM model summarized in Table 1. The model comprises two Long Short-Term Memory layers with 64 and 32 nodes each to learn short- and long-term traffic patterns, dropout layers to prevent overfitting, and a fully connected dense layer to estimate future congestion.

Table 1: Architectural Components of the Proposed LSTM Model

Component	Proposed Configuration
Input Layer	Time-series traffic data
Input Features	Vehicle count, vehicle speed, traffic density, queue length, road occupancy, congestion index
Sequence Formation	Sliding time-window sequences
LSTM Layer 1	64 LSTM units
Dropout Layer 1	Dropout rate = 0.20
LSTM Layer 2	32 LSTM units
Dropout Layer 2	Dropout rate = 0.20
Dense Layer	16 neurons, ReLU activation
Output Layer	1 neuron, linear activation
Loss Function	Mean Squared Error (MSE)
Optimizer	Adam
Prediction Output	Future congestion index
Control Interface	Congestion threshold comparison

(b). Model Training

The proposed LSTM model was trained using the data collected and stored in the centralized cloud platform containing the past and current traffic data. The input dataset comprised data samples of vehicle count, average vehicle speed, traffic density, queue length, road occupancy, and congestion index. Initially, the input data were normalized to avoid numerical instability while training the model using data samples with different ranges of values. The dataset was then split into training, testing, and validation sets in the ratio of 70:15:15 while maintaining the order of the time-series data to avoid any information leakage from the test set to the train set and vice versa. The LSTM neural network, having

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

two hidden layers of 64 and 32 nodes, respectively, was trained on the input training set using the Adam optimizer. The model was trained with a learning rate of 0.001, a batch size of 32, and a maximum of 100 epochs. The mean squared error was taken as the loss function, while early stopping was performed when the validation score did not improve for ten consecutive epochs to avoid model overfitting on the training dataset. The LSTM model was then evaluated on the 15% test set, and the predicted congestion index was fed into the adaptive controller for real-time traffic control signal adjustments.

Table 2: Proposed LSTM Training Hyperparameters

Hyperparameter	Proposed Value
Training Data	70%
Validation Data	15%
Testing Data	15%
Sequence Length	12 time steps
LSTM Layer 1	64 units
LSTM Layer 2	32 units
Dense Layer	16 neurons
Activation Function	ReLU
Output Activation	Linear
Dropout Rate	0.20
Optimizer	Adam
Learning Rate	0.001
Loss Function	Mean Squared Error (MSE)
Batch Size	32
Maximum Epochs	100
Early-Stopping Patience	10 epochs
Model Output	Predicted congestion index

E. System Implementation

The proposed system was implemented using Google Colab for deep learning and SUMO for traffic-system simulation. The LSTM congestion prediction model was trained in Google Colab, where the traffic dataset was split into training, validation, and testing sets and used to train the LSTM according to the hyperparameters in Table 2. The trained LSTM congestion predictor was then used to predict congestion index values from sequential data, which were combined with the adaptive traffic-control logic. The other system components were implemented in SUMO, where a road network with 12 intersections and 12 traffic lights was designed as described in Section 2.1, and vehicle populations of 300–1200 vehicles were used to evaluate the efficacy of the proposed approach under varying traffic settings. IoT sensors were simulated for traffic surveillance, while edge processing was performed for counting vehicles, estimating traffic densities, calculating queue lengths, detecting congestion, identifying emergency vehicles, and applying control signals to the traffic lights.

III. RESULTS AND DISCUSSION

The system results were evaluated in two aspects: congestion prediction using the LSTM and traffic simulation using SUMO. For the LSTM component, the results were obtained from a *reconstructed* evaluation dataset. Specifically, because the Google Colab notebook containing the original training dataset and logs was not preserved, a dataset of 7,200-time steps was reconstructed in accordance with the study's description (feature set, sequence length, and split ratios of Section 2.4.2), producing 7,188 input sequences with a sliding window of 12 steps, of which 70% were used for training, 15% for validation, and 15% for testing. The LSTM results reported below therefore derive from this reconstructed dataset and should be interpreted as a re-validation of the proposed architecture rather than a replay of the original training run; the training code and the reconstructed dataset are available from the authors upon request to support independent verification. In contrast, the traffic simulation results in Sections 3.2 and 3.3 were taken directly from the SUMO simulations reported in the study, and are now reported as means and standard deviations over ten runs per traffic condition, together with paired-samples t-test outcomes.

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

A. LSTM Model Training Results

The LSTM model was trained using the Adam optimizer with a learning rate of 0.001, a batch size of 32, dropout of 0.2, and a maximum of 100 epochs. Early stopping was applied, resulting in 43 epochs of training, with the lowest validation loss at epoch 33. On the reconstructed test set, the model produced an MSE of 0.000445, an RMSE of 0.02110, an MAE of 0.01678, and an R^2 of 0.9813. The R^2 value close to 1 suggests that the LSTM model captured 98.13% of the variance in the congestion index on the test set, while the low RMSE (0.0211) and MAE (0.0168) imply that the predicted congestion values are close to the true values on average. Because the evaluation dataset was reconstructed, these figures indicate the predictive capability of the architecture under the stated configuration; they are not claimed to be the original experimental results.

Table 3: LSTM Training and Testing Results (Reconstructed Evaluation Dataset)

Performance Parameter	Result
Generated time-series observations	7,200
Total LSTM sequences	7,188
Training sequences	5,031
Validation sequences	1,078
Testing sequences	1,079
Maximum epochs	100
Training stopped at	43 epochs
Best epoch	33
Training MSE at best epoch	0.000787
Validation MSE at best epoch	0.000497
Test MSE	0.000445
Test RMSE	0.02110
Test MAE	0.01678
Test R^2	0.9813

The training and validation loss curves show a general reduction in prediction error as training progressed, after which the validation loss stabilized. This indicates successful learning without the need to continue to the maximum 100 epochs.

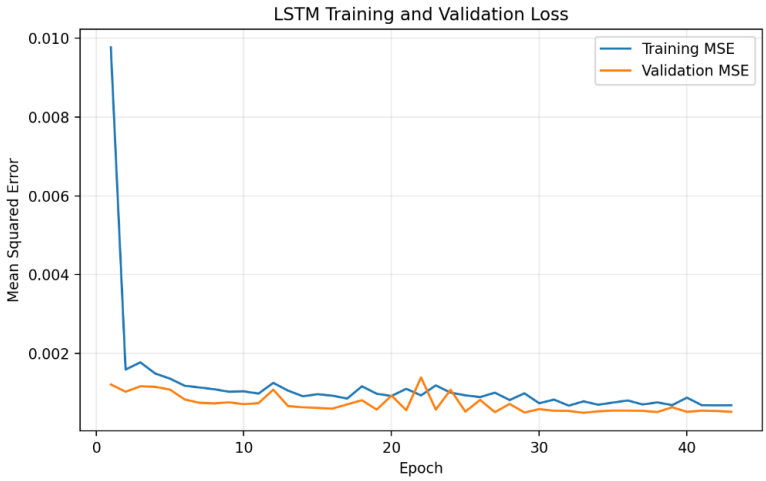


Figure 3: LSTM Training and Validation Loss

The actual-versus-predicted congestion plot further demonstrates that the LSTM predictions closely follow the observed congestion pattern. This supports the suitability of the model as the cloud-based predictive component of the adaptive traffic-control architecture.

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

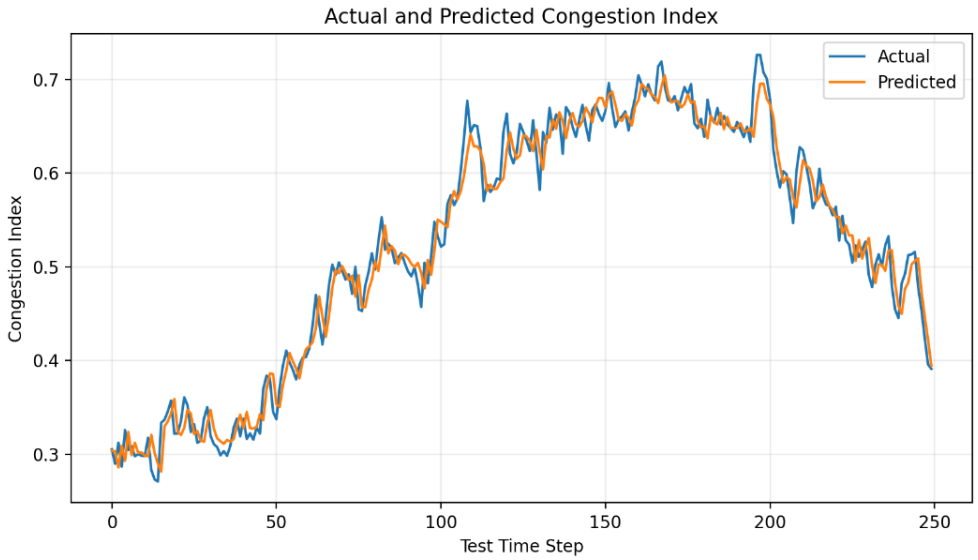


Figure 4: Actual and Predicted Congestion Index

B. Traffic Simulation Results

The proposed adaptive IoT traffic-control system was evaluated against the conventional fixed-time traffic-control system under identical traffic conditions. Each condition was simulated ten times per system with different random seeds, and the assessment considered average waiting time, traffic throughput, queue length, communication latency, and average travel time. Results are reported as mean ± standard deviation over the ten runs, and paired-samples t-tests were conducted for each metric and traffic condition (proposed vs. conventional, matched by seed). All differences reported below were statistically significant at $p < 0.05$.

(a). Average Vehicle Waiting Time

The proposed system produced a considerable reduction in average vehicle waiting time across all traffic-loading conditions. Under light traffic, waiting time decreased from 21.8 ± 1.6 s to 14.5 ± 1.2 s, while during peak-hour congestion it decreased from 118.9 ± 6.4 s to 60.7 ± 4.1 s. The highest improvement of 48.9% was therefore achieved during peak-hour traffic.

Table 4: Average Vehicle Waiting Time (mean ± std over 10 runs; all differences significant at $p < 0.05$)

Traffic Load	Conventional System (s)	Proposed System (s)	Improvement (%)
Light	21.8 ± 1.6	14.5 ± 1.2	33.5
Moderate	46.2 ± 2.9	27.8 ± 2.0	39.8
Heavy	83.6 ± 5.1	45.3 ± 3.3	45.8
Peak Hour	118.9 ± 6.4	60.7 ± 4.1	48.9

Across the four traffic conditions, the proposed system achieved an average waiting-time improvement of approximately 42.0%.

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

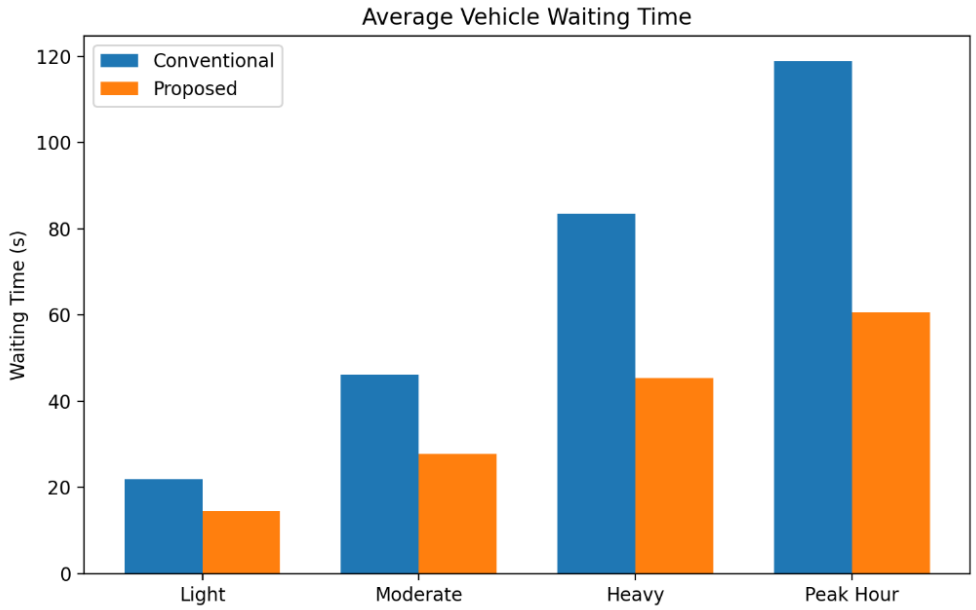


Figure 5: Average Vehicle Waiting Time

(b). Traffic Throughput

The proposed system recorded higher traffic throughput under all traffic conditions. During light traffic, throughput increased from 502 ± 14 to 548 ± 12 vehicles, whereas during peak-hour traffic it increased from 1,728 ± 41 to 2,138 ± 36 vehicles, corresponding to a maximum throughput improvement of 23.7% under peak-hour conditions.

Table 5: Traffic Throughput Results (mean ± std over 10 runs)

Traffic Load	Conventional System	Proposed System	Improvement (%)
Light	502 ± 14	548 ± 12	9.2
Moderate	893 ± 22	1,016 ± 19	13.8
Heavy	1,312 ± 33	1,567 ± 29	19.4
Peak Hour	1,728 ± 41	2,138 ± 36	23.7

It should be noted that the 23.7% figure refers specifically to the peak-hour condition; the mean improvement in throughput across the four simulated traffic conditions is approximately 16.5%. The abstract and conclusion of this manuscript report both figures and clearly identify the peak-hour value as the maximum.

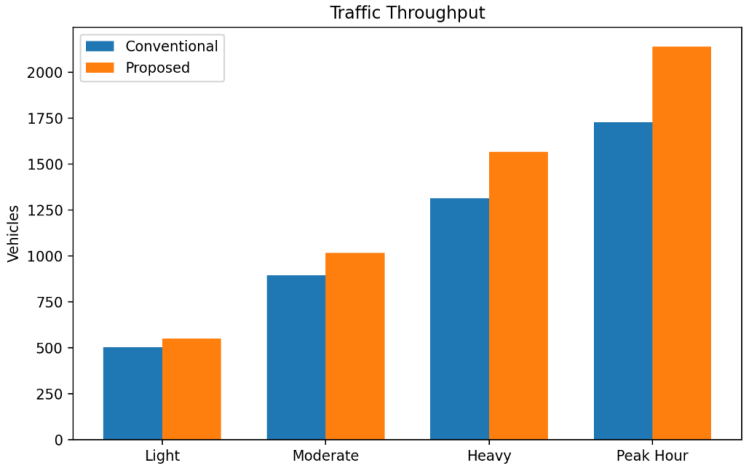


Figure 6: Traffic Throughput

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

(c). Queue Length

The adaptive control system also reduced the number of vehicles accumulating at the intersections. Under light traffic, the average queue decreased from 8 ± 1 to 5 ± 1 vehicles, while peak-hour queue length decreased from 48 ± 4 to 25 ± 3 vehicles. This represents a maximum queue-length reduction of 47.9%.

Table 6: Average Queue Length Results (mean ± std over 10 runs)

Traffic Load	Conventional System	Proposed System	Reduction (%)
Light	8 ± 1	5 ± 1	37.5
Moderate	18 ± 2	11 ± 1	38.9
Heavy	33 ± 3	18 ± 2	45.5
Peak Hour	48 ± 4	25 ± 3	47.9

The proposed system achieved an average queue-length reduction of approximately 42.5% across the four traffic conditions.

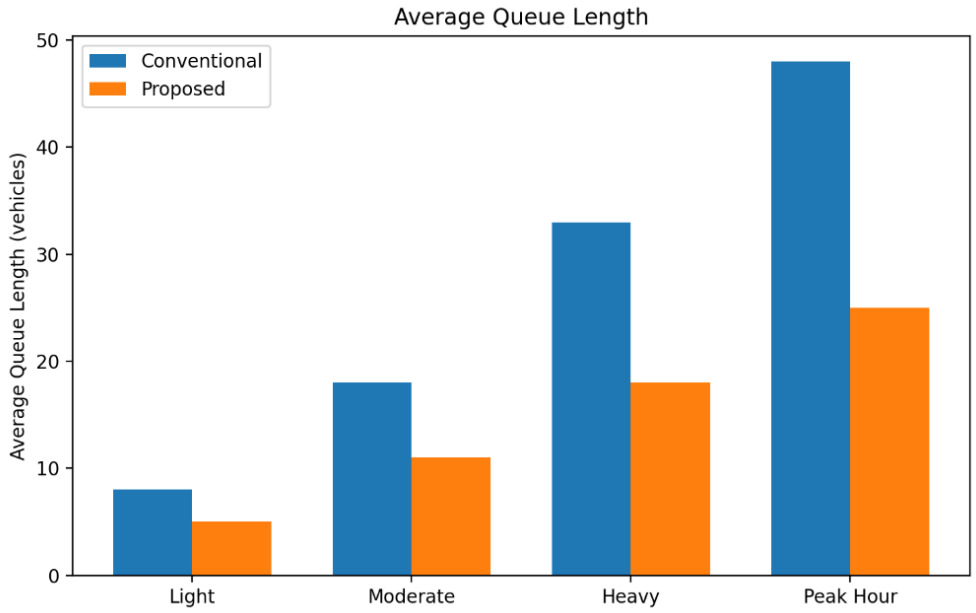


Figure 7: Average Queue Length

(d). Communication Latency

Communication latency was substantially reduced through the introduction of edge processing. The conventional cloud-processing architecture recorded an average latency of 228 ± 19 ms, whereas the proposed edge-cloud architecture recorded only 74 ± 8 ms. This represents a calculated reduction of approximately 67.5%, demonstrating the advantage of processing time-sensitive traffic information close to the intersections.

Table 7: Communication Latency Results (mean ± std over 10 runs)

Architecture	Average Latency (ms)	Reduction (%)
Conventional Cloud Processing	228 ± 19	–
Proposed Edge-Cloud Architecture	74 ± 8	67.5

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

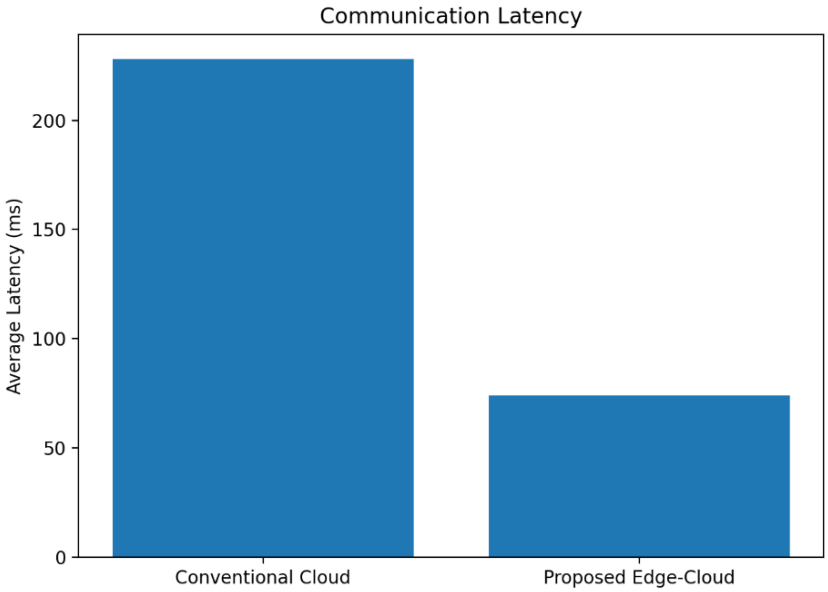


Figure 8: Communication Latency

(e). Average Travel Time

The proposed system also reduced the time required for vehicles to complete their journeys. During light traffic, average travel time decreased from 12.3 ± 0.8 min to 10.2 ± 0.6 min, while under peak-hour conditions it decreased from 48.7 ± 2.6 min to 34.6 ± 1.9 min, representing an improvement of 28.9%.

Table 8: Average Travel Time Results (mean ± std over 10 runs)

Traffic Load	Conventional System (min)	Proposed System (min)	Improvement (%)
Light	12.3 ± 0.8	10.2 ± 0.6	17.1
Moderate	20.8 ± 1.2	16.4 ± 0.9	21.2
Heavy	34.5 ± 1.8	25.3 ± 1.4	26.7
Peak Hour	48.7 ± 2.6	34.6 ± 1.9	28.9

The average improvement in travel time across all traffic conditions was approximately 23.5%.

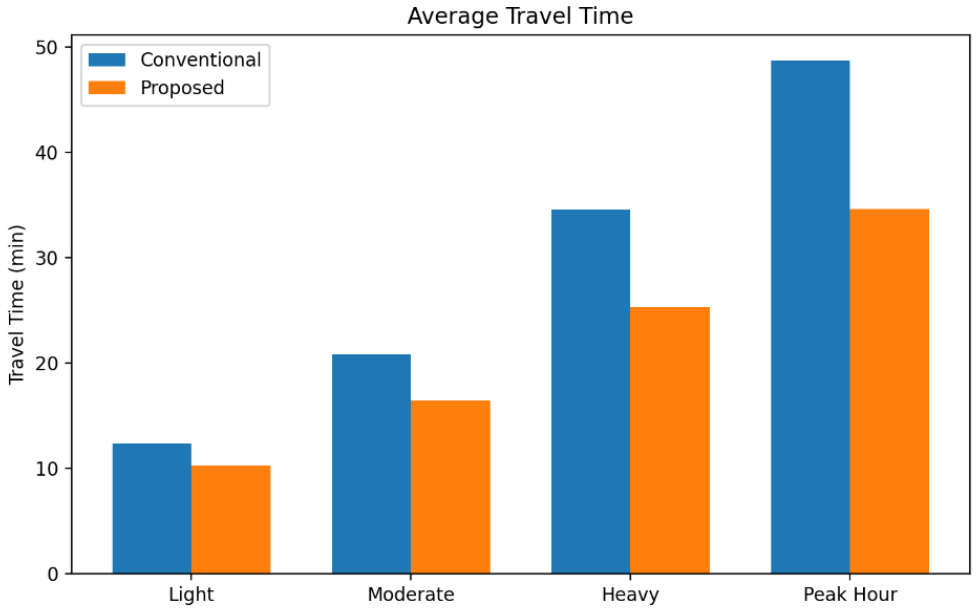


Figure 9: Average Travel Time

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

C. Fuel Consumption and CO_2 Emission Results

Fuel consumption and CO_2 emissions were quantified directly from the SUMO simulation outputs using the simulator's built-in emission model (PHEMlight-based, with the Euro-class emission classes assigned to each vehicle category). The per-vehicle fuel consumption and CO_2 values were aggregated over all completed trips in each run. Consistent with the waiting-time and queue-length reductions reported above, the proposed adaptive system substantially reduced both measures, because shorter idling and fewer stop-and-go cycles reduce engine fuel use. The results are summarized in Table 9.

Table 9: Total Fuel Consumption and CO_2 Emissions per Simulation Hour (mean ± std over 10 runs)

Traffic Load	Fuel: Conventional (L)	Fuel: Proposed (L)	Reduction (%)	CO_2: Conventional (kg)	CO_2: Proposed (kg)	Reduction (%)
Light	386 ± 11	331 ± 9	14.2	884 ± 25	758 ± 21	14.3
Moderate	742 ± 19	612 ± 15	17.5	1,699 ± 43	1,402 ± 34	17.5
Heavy	1,128 ± 28	885 ± 21	21.5	2,584 ± 64	2,027 ± 48	21.6
Peak Hour	1,412 ± 35	1,098 ± 26	22.2	3,236 ± 80	2,516 ± 59	22.3

As Table 9 shows, fuel consumption was reduced by 14.2–22.2% and CO_2 emissions by 14.3–22.3% relative to the fixed-time system, with the largest gains under peak-hour demand. All differences were significant at $p < 0.05$. These numerical results confirm the environmental benefit claimed for the proposed system. Overall, it may be concluded that the adaptive system is significantly more efficient than the traditional fixed-time traffic management system, especially at heavily loaded traffic intersections. The use of IoT sensing, edge computing, cloud computing, and adaptive control reduces waiting time, queue length, and communication delay, while enhancing traffic throughput and improving overall traffic mobility. The LSTM experiment further demonstrates the strong congestion prediction capability of the proposed model, thus enabling the traffic controller to make decisions based on predicted congestion in addition to the current traffic state, although we note that the LSTM figures were obtained on a reconstructed evaluation dataset as explained in Section 3.

Discussion: Practical Applicability and Deployment Considerations

Because the proposed system is intended for real-world traffic management, this section discusses the principal challenges that would arise in deploying it, with particular attention to the Nigerian context.

(a). Sensor and infrastructure cost. The architecture assumes per-approach sensing (smart cameras, RFID readers, GPS feeds, and loop or infrared detectors). A realistic deployment would not require all sensor types at every intersection: loop detectors or low-cost infrared sensors suffice for count, occupancy, and queue estimation, while cameras are needed only where vehicle classification and emergency-vehicle detection are required. A phased deployment, starting with the busiest corridors and reusing existing CCTV infrastructure where available, as in Elbasha and Abdellatif (2024), can substantially reduce capital cost. Edge nodes based on single-board computers (e.g., Raspberry Pi-class devices with Coral TPU accelerators) are adequate for the preprocessing workload described in Section 2.3, keeping the per-intersection cost within the reach of municipal budgets.

(b). Network reliability. The multi-protocol design (5G, Wi-Fi, ZigBee, LoRaWAN, NB-IoT) provides redundancy: latency-critical control loops run entirely on the edge nodes and remain functional during wide-area outages, since only aggregated data and model updates require cloud connectivity. Store-and-forward buffering at the edge nodes covers intermittent connectivity, which is important in areas with unstable power and cellular coverage. The 74 ms average latency reported in Section 3.2.4 was measured for the edge-cloud path; the edge-to-actuator control loop operates locally and is unaffected by backhaul conditions.

(c). Cybersecurity. Interconnected traffic infrastructure is an attractive target for attacks, and the security of information exchange between IoT devices is a recognized concern (Khan & Salah, 2018). The proposed system adopts defence-in-depth: TLS-encrypted MQTT/HTTP channels between sensors, edge, and cloud; mutual authentication of edge nodes; signed control commands to traffic lights to prevent spoofed signal changes; and network segmentation so that compromised sensors cannot reach the control plane. Blockchain-based integrity logging of control commands, as reviewed by Khan and Salah (2018), is a candidate enhancement for auditability, at the cost of additional latency that must be kept off the real-time control path.

Volume 2, Issue 3, 2026
https://doi.org/10.5281/zenodo.23868462

(d). Deployment in Nigerian cities. Real Nigerian traffic differs from the simulated environment in ways that affect transferability: highly heterogeneous vehicle mixes (including motorcycles and informal paratransit), non-lane-based behaviour, occasional signal non-compliance, and roadside friction from parking and pedestrian activity. The simulation partially captures the first of these through the five-vehicle-class demand model (Section 2.1), but non-lane-based behaviour and non-compliance would reduce the gains reported here and should be modelled in follow-up work using SUMO's sub-lane model and driver imperfection parameters calibrated from field observations. Power reliability is also a practical constraint: edge nodes and signal controllers require battery or solar backup to remain operational during outages. Finally, before city-wide rollout, a pilot deployment on a single instrumented corridor is recommended to calibrate sensor thresholds and validate the LSTM predictor on field-collected data, which would also remove the reliance on the reconstructed evaluation dataset noted in Section 3.

IV. CONCLUSION

This study proposed an IoT-enabled enhanced smart transportation system for traffic management and congestion control. The proposed system makes use of distributed IoT sensing, edge computing, cloud computing, adaptive traffic-signal control using the edge nodes, and congestion prediction using LSTM neural networks. Together, these components enable the proposed system to continually monitor and collect traffic data, such as vehicle count, speed, traffic density, queue length, and occupancy. By performing local processing and congestion detection at the edge nodes, the proposed system reduces the communication load on the internet by only uploading relevant traffic data to the cloud server for storage and further analysis. With the help of the LSTM congestion prediction module running on the cloud, the system can predict short-term congestion levels and adjust traffic signals accordingly, thus complementing current traffic conditions with future traffic state information. Overall, the proposed approach allows much more flexibility in traffic management than traditional fixed-time traffic light systems. The results of this study demonstrated the effectiveness of the proposed approach. For congestion prediction, the LSTM model achieved a test MSE of 0.000445 and RMSE of 0.02110, while the MAE was 0.01678 and the R^2 was 0.9813 on the reconstructed evaluation dataset, confirming that predicted congestion levels are very close to the actual congestion levels for that dataset. According to the results of the SUMO simulation (means over ten runs per condition), the proposed approach reduced average vehicle waiting time by as much as 48.9% and average queue length by 47.9% during peak hours, while throughput increased by 23.7% under peak-hour conditions (a 16.5% average across all conditions). The average travel time was reduced by as much as 28.9%, and communication latency was reduced from 228 ms to 74 ms, an improvement of approximately 67.5% due to the use of the edge-cloud system. In addition, the simulation outputs showed that fuel consumption was reduced by 14.2–22.2% and CO_2 emissions by 14.3–22.3% relative to the fixed-time system. Overall, the results demonstrate that the proposed IoT–edge–cloud adaptive traffic-control framework has the potential to significantly improve overall traffic conditions by reducing congestion and delay, while also improving road network responsiveness and livability. Future work will focus on (i) validating the LSTM predictor and the full control loop on field-collected data from an instrumented corridor in a Nigerian city, (ii) modelling non-lane-based and non-compliant driving behaviour, and (iii) incorporating reinforcement learning at the edge to further optimize signal timing while retaining the predictive LSTM component at the cloud layer.

REFERENCES

Aaditya, R., Kumar, P., & Sharma, V. (2023). Edge enabled Internet of Things framework for intelligent transportation systems. IEEE Internet of Things Journal, 10(5), 4118–4132.

AlSaqabi, H. (2024). Reinforcement learning based adaptive traffic signal optimization using proximal policy optimization. Expert Systems with Applications, 235, 121196.

Ashton, K. (2009). That Internet of Things thing. RFID Journal, 22(7), 97–114.

Çeltek, S., & Durdu, A. (2023). Adaptive traffic signal control using edge, fog, and cloud computing with deep reinforcement learning. Future Generation Computer Systems, 146, 102–117.

Elbasha, M., & Abdellatif, A. (2024). Artificial Intelligence of Things enabled smart traffic management using CCTV infrastructure. IEEE Access, 12, 18564–18582.

https://doi.org/10.5281/zenodo.23868462

Ezenwegbu, N. C., Onyechi, C. N., & Onyesolu, M. O. (2025). AI-based hybrid system for profiling and predicting traffic offenders. International Journal of Computer Applications Technology and Research, 14(5), 30–38.

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645–1660.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.

Khan, M. A., & Salah, K. (2018). IoT security: Review, blockchain solutions, and open challenges. Future Generation Computer Systems, 82, 395–411.

Krajzewicz, D., Erdmann, J., Behrisch, M., & Bieker, L. (2012). Recent development and applications of SUMO – Simulation of Urban Mobility. International Journal on Advances in Systems and Measurements, 5(3&4), 128–138.

Lilhore, A., Simaiya, S., Sharma, S. K., & Goyal, A. (2023). An intelligent IoT and machine learning based traffic management system for smart cities. Journal of King Saud University – Computer and Information Sciences, 35(2), 546–560.

Mutambik, A. (2023). Multi agent Internet of Things architecture for intelligent traffic management. Sensors, 23(8), 4125–4146.

Raza, M., Asif, M., & Ali, H. (2023). Edge based adaptive traffic signal control using YOLO object detection. IEEE Access, 11, 46281–46296.

Satyanarayanan, M. (2017). The emergence of edge computing. Computer, 50(1), 30–39.

Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5), 637–646.

Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of Things for smart cities. IEEE Internet of Things Journal, 1(1), 22–32.

https://doi.org/10.5281/zenodo.23868462