LSTM-Based QoS-Aware Proactive Flow-Rule Placement in Software-Defined Internet of Vehicles

P. Sowmya
Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Bengaluru, Karnataka, India
sowmya-ece@dayanandasagar.edu

P. Dinesha
Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Bengaluru, Karnataka, India
drdinesh-ece@dayanandasagar.edu (corresponding author)

Anitha Saravana Kumar
Olsen College of Engineering and Science, Fairleigh Dickinson University, Vancouver, Canada
a.saravanakumar@fdu.edu

J. Akilandeswari
Department of Information and Technology, Sona College of Technology, Salem, India
akilandeswari@sonatech.ac.in

Received: 13 February 2026 | Revised: 23 March 2026 | Accepted: 10 April 2026

 DOI: https://doi.org/10.48084/etasr.18164

ABSTRACT

With the rapid growth of mobile devices and bandwidth-intensive applications in the Internet of Vehicles (IoV), traffic and flow management among vehicular nodes poses a challenge to routing efficiency. In Software Defined Networks (SDN), the control plane is abstracted from the data plane, providing a centralized, programmable network control. The default flow-rule placement in SDN, being reactive, leads to suboptimal network performance, such as latency issues and packet loss. Therefore, there is a need for intelligent, proactive, and adaptive network traffic management systems to maximize network performance. This study introduces a Long Short-Term Memory (LSTM)-based Quality of Service (QoS)-aware proactive flow-rule placement in Software Defined Internet of Vehicles (SDIoV). First, by exploiting trends in vehicle movements, an LSTM model is trained to forecast the next-step Access Point (AP) linked to vehicles. Second, the SDN controller proactively installs QoS-aware flow rules in the predicted AP for classified traffic to meet its QoS requirements. These flow rules dynamically reroute traffic using OpenFlow rules and allow execution of incoming request actions without the controller's intervention. The proposed scheme demonstrates superiority over existing methods by combining SDN with machine learning to enable proactive, adaptive, and efficient traffic management in dynamic IoV networks.

Keywords-Software Defined Networks (SDN); Internet of Vehicles (IoV); Long Short-Term Memory (LSTM); flow-rules; routing; OpenFlow

I. INTRODUCTION

In the Internet of Vehicles (IoV), the growing demand for digital communication, cloud-based services, and high-definition multimedia streaming has caused a significant rise in real-time network traffic. This explosive growth poses a complex challenge: managing data flow while ensuring Quality of Service (QoS) by minimizing latency and enhancing reliability [1]. Conventional traffic management strategies, often based on static rules and manual configurations, are inefficient in coping with dynamic, high-volume modern networks. Optimal data transmission requires QoS, which existing Internet infrastructures cannot promise [2]. Consequently, there is a critical need for more intelligent, adaptive, and scalable traffic control mechanisms.

Various IoV applications necessitate distinct network QoS assurances, including throughput, latency, jitter, packet drop, and reliability. Their diverse applications require substantial computing resources to meet the rigorous demands of real-time users. Vehicles encounter difficulties in achieving performance standards due to resource limitations [3]. Software-Defined Networks (SDN) are essential to meet the evolving demands of IoV [4]. SDN separates data and control planes, streamlining network management and facilitating integrated network control. A centralized controller abstracts control logic from networking devices, rendering them as mere high-speed forwarding elements. Vendor-specific configurations across diverse devices limit the usability of IoV. Hardware devices must be configured instantaneously to meet application-specific requirements. Separation of the control plane from conventional hardware in SDN allows real-time reconfiguration of network devices at the application level to satisfy application-specific demands, independent of low-level hardware configurations [5]. Therefore, SDN is regarded as a solution for the dynamic QoS requirements of IoV applications.

In IoV settings, vehicles connect to the network through wireless Access Points (AP). APs exchange user and backbone network information through flow tables, which are centrally managed by the SDN controller, and decide how packets are handled by network devices. Flow-table rules need to be optimally managed based on vehicle movement and requests in the vicinity of the associated AP. Existing SDN-based reactive approaches fail to provide an optimal flow-rule placement scheme that considers the mobility constraints of vehicles and the capacity constraints of APs. In a reactive approach, for new requests, the SDN controller places the flow rules on the AP only after receiving Packet-In messages from the AP [6]. The mobility of vehicles requires the flow rules to be updated frequently, in turn generating more Packet-In messages to the controller. This results in slow data transfer, packet loss, and delay, increasing network overhead. Alternatively, in proactive approaches, the controller proactively installs the flow rules in the APs based on vehicle movements in the network. The proposed method adopts a proactive scheme to address the time-criticality and drop constraints in IoV.

APs in SDN networks support devices with different QoS demands. Differentiated services provide a QoS framework by classifying traffic and marking packets with the 6-bit Differentiated Services Code Point (DSCP) field in the IP header to map traffic to reasonable QoS levels, enabling differentiated flow treatment across network devices [7]. In addition, deep learning models, such as Long Short-Term Memory (LSTM), play a significant role in leveraging the potential of SDNs to anticipate network behavior. LSTM networks are highly effective in modelling sequential and time-series data [8] and are ideal for predicting network traffic patterns and enabling proactive network management. Thus, deep learning techniques can be used with an SDN controller to improve traffic and reduce congestion.

In [1], the incoming flows were divided into sub-flows based on bandwidth, routed individually into multiple paths, and the best paths were selected using a greedy heuristic approach. In [9], an Asynchronous Federated Deep Reinforcement Learning (AFDRL) approach was proposed for Vehicular Edge Computing (VEC) environments to examine computation and queue models for task execution at the vehicles. The study in [10] aimed to improve the performance of various QoS metrics, such as the end-to-end delay and packet delivery ratio, in vehicular networks facing challenges such as high mobility, timely and reliable delivery, and frequent topology changes, introducing an optimized routing protocol to manage and enhance the QoS of flows for varying node sizes. In [11], four architectural approaches to IoT networks for smart city applications were proposed, and critical network QoS metrics were discussed. This study concluded that delay and packet loss are important QoS metrics that must be considered at the network layer, but the discussion mainly focused on high-level architectural issues for QoS requirement analysis.

This study integrates an LSTM-based proactive AP prediction model with proactive flow-rule placement to make better routing decisions in Software Defined Internet of Vehicles (SDIoV) networks. The proposed flow-rule distribution policy classifies traffic based on DSCP values and proactively installs flow-rules at the predicted AP. This alleviates controller intervention and enhances QoS under diverse traffic loads. In addition, the SDN controller classifies flows as time-critical, drop-critical, and best-effort, and individually assigns them to distinct queues based on QoS requirements, thus ensuring reduced delay and improved QoS and throughput for real-time heterogeneous networks. The problem is challenging as vehicles move between locations, and the flow rules must dynamically change in APs. The proposed dynamic adaptive traffic management system for SDIoV networks, integrating an LSTM-based prediction model to forecast AP selection in real time, enhances network efficiency, reduces latency, and minimizes packet loss.

II. SYSTEM MODEL

This section describes the system model of the proposed LSTM-based QoS-aware SDIoV network architecture. It focuses on a network traffic management system where an LSTM model predicts AP selection, and the SDN controller further proactively places the flow-rules at the predicted AP and also dynamically re-routes QoS classified traffic and handles queue management using SDN principles.

A. Network Architecture

Figure 1 illustrates the proposed SDIoV architecture, which comprises vehicles, APs, and an SDN-based network. This SDIoV model involves heterogeneous application-oriented vehicles that communicate with the AP. Based on the vehicle's movement history, the SDN controller proactively predicts its next associated AP, in which the flow rules are then proactively installed. The controller dynamically modifies the flow rules in the AP according to the QoS requirements and the anticipated locations of the users. Based on QoS specifications, flows are further categorized and forwarded to different queues. With the revised flow rules, the AP executes the necessary actions for the incoming user data.

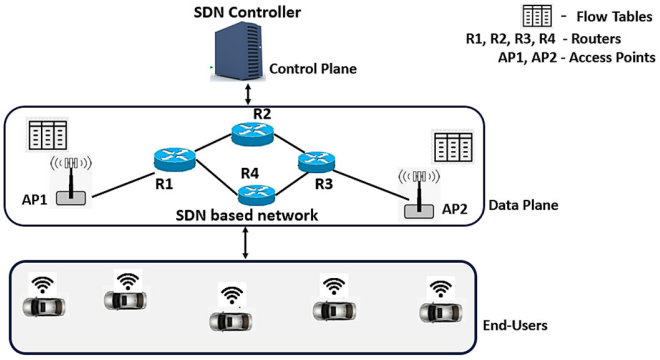

Fig. 1. SDIoV architecture.

B. SDN Flow Rules and Flow Tables

Unlike traditional networks, where packet forwarding is based on routing tables and distributed logic, SDN relies on flow tables that are centrally managed by the SDN controller. Flow tables are the fundamental components of the data plane and contain a set of rules installed on SDN-enabled devices, such as APs. Each rule plays a crucial role in deciding how packets are handled by these network devices.

Flow tables serve as the execution layer of the SDN architecture. On the arrival of a packet at an AP, the latter consults its flow table to determine the packet forwarding method. The AP first examines the incoming packet and attempts to find a matching entry in the flow table. Upon finding a match, the AP forwards the packet without consulting the controller. If no match is found, the packet is typically sent to the controller using a Packet-In message for further instructions. Once a flow rule is installed, all matching packets are processed in the AP, reducing latency and load on the controller. The controller can dynamically install or remove flow entries based on network policies, application needs, or traffic patterns. Flow tables enable fine-grained control over traffic flows, supporting advanced use cases such as QoS enforcement, security policies, and load balancing. By proactively or reactively managing flow entries, SDN can handle large-scale networks with dynamic traffic efficiently.

The controller installs proactive flow rules in advance for known traffic patterns, reducing intervention. Reactive flow rules are installed in response to a Packet-In event when no rule is matched. However, although this approach provides flexibility, it introduces latency. In a QoS-aware SDIoV network, a flow-rule may direct time-critical traffic to a low-latency path, and drop-critical traffic might be routed through a high-reliability path.

C. Requirement of Software -Defined AP

The SDN controller centrally controls the AP, which forwards traffic to the network according to the controller-defined flow rules. With the varying geographical locations of moving vehicles, their association with the AP also fluctuates. Therefore, an appropriate AP is required to be selected, among the ones available in the vicinity of the vehicle, to minimize the delay incurred in packet delivery and to avoid unnecessary utilization of the available AP resources to update the flow rules. Therefore, an adaptive, dynamic, and proactive scheme is crucial to predetermine the associated AP in terms of the delay incurred and efficient management of the resources. Typically, vehicles make their association decisions with APs. APs do not have any centralized control over the association, but can be controlled centrally with an agent. This study considers agent-configured APs so that the association of vehicles can be decided based on the APs' resource constraint capabilities. An SDN controller manages the networking devices in a centralized manner, leveraging a global view of the network.

D. LSTM Prediction Model

Given the resource-constrained nature of an AP, updating flow rules across all APs in the vehicle's vicinity consumes resources redundantly. Therefore, the SDN controller implements proactive prediction of the vehicle's future associated AP to store the flow-rules. LSTM, a type of Recurrent Neural Network (RNN), excels at modelling temporal dependencies, making it ideal for forecasting time-series data such as network traffic. It takes in historical and real-time vehicular data, such as position, speed, and direction, and predicts the AP next likely to be associated with the vehicle. Integrating an LSTM prediction model with SDN principles, the controller can make real-time decisions based on a data-driven approach that improves QoS and reduces latency and loss in SDIoV. LSTM networks outperform standard RNNs as they solve the vanishing gradient problem. Although traditional RNNs struggle with long-term dependencies, LSTM utilizes specialized memory cells and gating mechanisms to retain information over much longer sequences, making them ideal for time-series forecasting, such as network traffic. Compared to highly resource-intensive Convolutional Neural Networks (CNNs), which require more parameters for classification, LSTMs are lightweight and better suited for sequential data, resulting in lower computational load and faster execution at the SDN controller. This makes LSTMs the preferred choice for a real-time proactive prediction model in dynamic high-density environments.

E. DSCP-Based Traffic Classification

Using the DSCP field of the IP header, incoming traffic is classified into three QoS classes. In the data plane, each AP is associated with three predefined queues, ensuring that traffic is forwarded according to its QoS needs. Table I shows the mapping. When a packet arrives at an AP, the packets are checked for DSCP value match and forwarded to the queues accordingly. This facilitates the network to dynamically prioritize traffic while supporting unknown flows.

TABLE I. MAPPING OF DSCP VALUE TO QUEUES

DSCP	QoS Traffic	Allocated Queue
46	Time-critical	Queue 1
34	Drop-critical	Queue 2
0	Best-effort	Queue 3

F. Algorithm

Figure 2 illustrates the proposed SDN controller architecture. First, the LSTM Prediction Module collects vehicular data from the associated APs and predicts the vehicle's next AP based on its movement. The predicted AP is

communicated to the QoS-Aware Routing Module, as detailed in Algorithm 1, which then determines the proactive flow rules to be installed in the predicted AP at designated time intervals. The OpenFlow protocol facilitates router communication [12]. The QoS-Aware Flow Classifier and Adaptive Flow-rule Installer modules assist the QoS-Aware Routing Module in installing adaptive QoS-aware flow rules. The QoS-Aware Flow Classifier classifies the traffic into time-critical, drop-critical, and best-effort flows, as detailed in Algorithm 2. The Adaptive Flow-Rule Installer installs proactive or reactive rules in the predicted AP within a specified time window. This alleviates the controller's load and reactive handling overhead. To enforce priority, the QoS Queue Handler allocates distinct output queues to various classes of traffic.

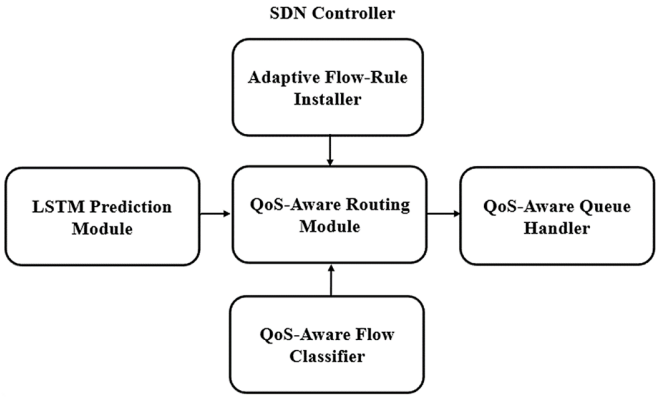

Fig. 2. Proposed SDN controller architecture.

```
Algorithm 1: LSTM-based AP prediction and
  controller communication.
Input: Dataset of vehicles D, time steps
  T_s, future step F_s
Output: Predictions of associated AP for
  each vehicle
procedure DATA-PROCESS
  Load D from CSV
  Sort D by veh_id and timestamp
  Normalize veh_id to [0,1]
  Standard features: position, speed,direction
end procedure

procedure SEQ-CREATION
  for each unique veh_id in D do
    Sliding windows extraction of length
    T_s for input
    Store AP at t + F_s as label
  end for
end procedure

procedure TRAIN-MODEL
  Encode AP IDs to integer classes
  Split into test sets
  Define the LSTM model
  Train with early stepping on validation
  loss
end procedure
procedure EVALUATE
```

Algorithm 1 illustrates the process for predicting a vehicle's future associated AP to install the flow-rules, using an LSTM-based model at the controller. The dataset of vehicles is loaded from a CSV file, sorted by the vehicle's ID and timestamp for time-based consistency. Vehicle IDs are normalized to the range [0, 1], while their features, such as position, speed, and direction, are standardized to have zero mean and unit variance to improve training convergence. Normalization ensures balanced feature contribution, stable gradients, and faster training convergence.

For each vehicle, sliding windows of length T_s are extracted, constituting the input sequences for the LSTM model. The target label for every sequence is the AP to which the vehicle connects after a future step F_s. AP IDs are encoded into integer classes to allow classification. The dataset is split into training and testing subsets. To prevent overfitting, training is conducted with early stopping using validation loss as the termination criterion. The trained model is evaluated on the test set using classification accuracy. A confusion matrix is generated to assess the model's ability to distinguish between AP classes, providing insight into faulty classification sets. With the vehicle ID and current time, the last T_s samples for the vehicle are fetched. If enough samples are available, features are scaled and fed into the LSTM model to produce a predicted AP ID; otherwise, the procedure returns none. The system iterates over all vehicles using a periodic infinite loop and predicts their future AP. If a prediction is available, a message containing the vehicle ID and predicted AP is sent to the QoS-Aware Routing Module. This enables proactive installation of the flow rule in the predicted AP.

When an AP connects to the controller, the default flow rules are deployed to maintain basic connectivity. The three default rules deployed are the ARP rule, the best-effort rule, and the table-miss rule. The ARP rule floods ARP packets to all ports to ensure proper address resolution in the network. For flows without special QoS guarantees, the default best-effort rule is deployed. To address unmatched flows, the table-miss rule triggers the controller to perform reactive handling, sending a Packet-In message to learn the flow and install a matching rule. The controller schedules proactive flow-rule installation in each AP in turns at specified time intervals. To prioritize preferential handling over best-effort traffic, these rules are installed with a higher priority. For packets that require low latency, the AP installs the time-critical rule, and for packets that demand high reliability and minimal packet loss, the drop-critical rule is installed. Time-critical traffic with DSCP 46 is then forwarded through the highest priority queue (Queue 1), and drop-critical traffic with DSCP 34 is forwarded through the medium priority queue (Queue 2). Best-effort traffic, DSCP 0, with no special QoS guarantees, is forwarded through the default lowest priority queue (Queue 3).

To ensure traffic priority, even when specific classifications are absent, the rule priorities are configured as shown in Table I. If an incoming packet matches a proactive rule, it is

forwarded directly to the corresponding queue based on its QoS requirements. Assuming that OpenVSwitch (OVS) queues have been pre-configured, the controller forwards the packets to the queues. This structured mechanism allows the network to dynamically prioritize traffic while supporting unseen flows. Algorithm 2 details the entire workflow. Upon packet arrival at the controller, the controller learns the source/destination MAC addresses and port numbers and installs the required reactive rule. The packet is now forwarded according to the standard policy. This mechanism reduces future controller intervention while adapting to previously unseen traffic.

```
Algorithm 2: DSCP-based QoS-aware
  Proactive Flow-rule Implementation in
  SDIoV networks.
Input: Predicted AP, Time window T seconds
  for proactive flow-rule installation,
  Flows with DSCP: 46 (Time-critical),
  34 (Drop-critical), 0 (Best-effort)
  Queue mapping: Queue1 → DSCP 46,
  Queue2 →DSCP 34, Queue3 → DSCP 0.
Output: QoS-aware reactive and proactive
  flow-rules installation on AP.
  DSCP-oriented QoS prioritization and
  Queue-oriented forwarding.
Default Flow-Rule Installation
  On AP connection:
    Load table-miss rule:
      match any → intimate controller
    Load best-effort rule:
      match DSCP = 0 → Queue 3,
      output: normal
Proactive Flow-Rule Installation
  At every T time interval,
     Load DSCP = 46 flow-rules → Queue 1
     Load DSCP = 34 flow-rules → Queue 2
     Ensure all QoS-aware flow rules are
     placed across all APs
Reactive Flow Handling
  On table-miss trigger
   Controller extracts src/dst MAC
   addresses
   DSCP Classification:
     DSCP = 46 → Queue 1,
     DSCP = 34 → Queue 2
     Load reactive rule: match src/dst →
     output: FLOOD
```

III. PERFORMANCE EVALUATION

A. Results and Discussion

The performance of the proposed method was assessed through extensive simulations and compared with benchmark algorithms such as FIFO and LRU, focusing on key metrics, including Round-Trip Time (RTT) and packet drops. Table II illustrates the simulation settings of the SDIoV network. The proposed method was evaluated using the Ryu SDN controller [13] and the Mininet network emulator [14]. Ryu is an open-source, Python-based SDN controller widely adopted for assessing OpenFlow-based SDNs. Table II shows the various simulation parameters considered. The proposed LSTM-based SDIoV was implemented on the TensorFlow platform (v2.16.1) [15]. Data were collected from the Caltrans Performance Measurement System (PeMS) [16]. The dataset used in this study is publicly available in [17]. All models were built using the Keras API.

TABLE II. SIMULATION SETTINGS

Parameter	Value
Number of APs	10
Number of Nodes	10
Speed of the Nodes	0-30 mph
Mobility model of AP	Static
Mobility model of Nodes	Flow-based

The LSTM model was designed for sequence processing. It starts with an LSTM layer that outputs sequences with 128 features over 10-time steps, followed by a dropout layer to reduce overfitting. A second LSTM layer compresses the representation to 64 features, followed again by dropout. The final dense layer produces 6 output values, likely corresponding to a classification task with 6 classes. The model has 118,918 trainable parameters and no non-trainable parameters. The hyperparameters were a learning rate of 0.0001 and a batch size of 64. Using the time series data, the LSTM model predicts the next AP for the vehicle. The input is a sequence of time steps with (x, y) coordinates, speed, direction, and normalized node ID features. Dropout is used to prevent overfitting. The LSTM model was trained with categorical cross-entropy loss and the Adam optimizer, using a validation split and early stopping to monitor and control overfitting.

Figure 3 illustrates the distribution of training target variables over time. The variables are bandwidth usage, average latency, and average packet rate. All three signals were normalized between 0 and 1. Highly variable values indicate dynamic behavior in the network data. Absence of a clear trend or seasonality suggests a noisy and complex dataset.

Figure 4 shows a confusion matrix for a multi-class classification model with 10 classes. Each row represents the actual class, and each column represents the predicted class. High values on the diagonal indicate right predictions, while off-diagonal values represent misclassifications. The color intensity helps visualize accuracy, with brighter squares showing higher values.

Figure 5 illustrates AP predictions for vehicle movement over a time span measured in minutes using an LSTM model. LSTM facilitates long-term learning by continuously updating the input within its memory structure. A pattern fluctuation occurs during training. As learning duration increases, the prior dataset helps the LSTM achieve higher prediction accuracy, demonstrating that it can effectively learn and retain long-term dependencies.

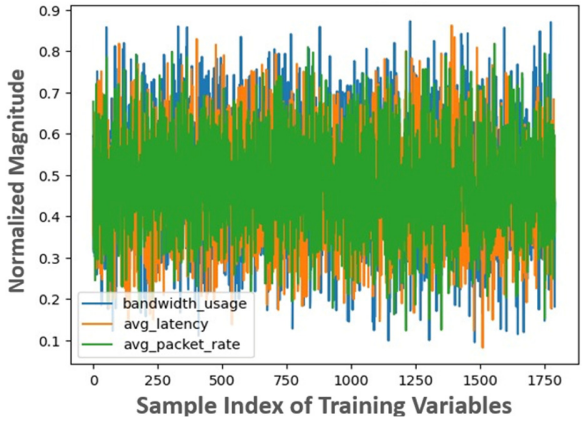

Fig. 3. Distribution of training targets.

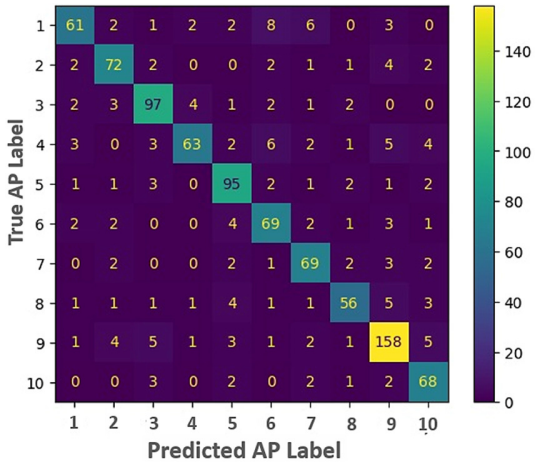

Fig. 4. A confusion matrix to evaluate the performance of AP predictions.

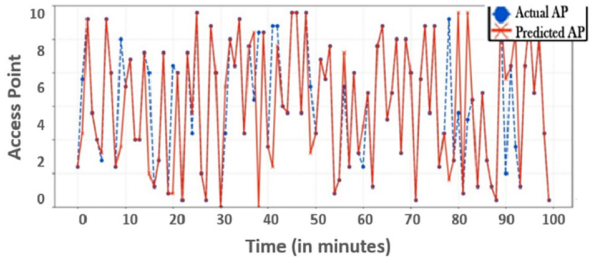

Fig. 5. AP prediction for vehicle movement using LSTM.

Table III presents the average RTT observed for each traffic type. Flow rules were installed at each AP according to the controller's internal timer. The system was tested for reactive and proactive rule installations using DSCP values. Results confirmed successful pre-installation of proactive flow-rules for DSCP 46 and DSCP 34 for time-critical and drop-critical traffic, respectively. For DSCP 0 best-effort traffic, which lacked a matching proactive rule, reactive rules were triggered. Based on DSCP values, traffic is mapped into three QoS queues. Queue counters were monitored to confirm correct classification and forwarding behavior for each traffic type. To evaluate the impact of controller mode, ICMP echo requests were sent, marking packets with DSCP 46 for time-critical and DSCP 34 for drop-critical traffic.

TABLE III. AVERAGE LATENCY FOR DIFFERENT TRAFFIC TYPES

Traffic type	Average latency (ms)
Time-critical	20.13
Drop-critical	32.21
Best-effort	3988.02

Figures 6 and 7 indicate that the model converges without severe overfitting. Figure 6 tracks the error rate, i.e., loss during the training process, where the lower is better. Both the training loss and validation loss start high at the beginning at 2.2425 and 2.1871. As the epochs progress, both lines decrease, showing that the model is learning effectively. Around epoch 10, a noticeable gap begins to form. The training loss continues to drop drastically to almost zero (0.0775), while the validation loss levels off, ending much higher at 0.7571. Figure 7 tracks the proportion of correct predictions the model makes, i.e., accuracy, where higher is better. Starting from low accuracies of around 17% to 20%, both metrics steadily climb as the model is being trained. Similar to the loss graph, the training accuracy climbs to a near-perfect score of 0.9803 (98.03%). However, the validation accuracy (orange) grows more slowly and plateaus, finishing at 0.8088 (80.88%).

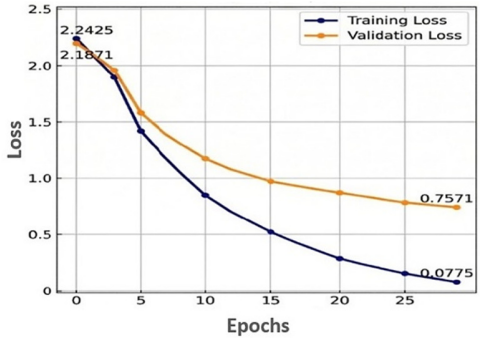

Fig. 6. Training curve for tracking loss.

Fig. 7. Training curve for tracking accuracy.

Figure 8 shows a sensitivity analysis that indicates how each input feature influences the model's output. The values of x and y positions of the vehicles show the strongest positive effect, with values of +0.18 and +0.09. The value of vehicle speed shows a negative effect (–0.12), indicating that higher speed affects performance. The direction of vehicle movement barely has any effect, and vehicle id also has a negligible negative effect (–0.01).

Figures 9 and 10 illustrate the results that compare the performance of the proactive and reactive controllers for the time-critical and drop-critical traffic, respectively, with proactive RTTs ranging on average from 0.15 ms for the former and 0.267 ms for the latter traffic. This indicates stable, consistent, low-latency, and predictable behavior. In contrast, the reactive controller experienced a high standard deviation of 2.59 ms and 2.274 ms for time-critical and drop-critical traffic. This delay and variability are due to the controller's processing time for new flows. Jitter and delay spikes can degrade QoS. These results reflect that proactive rule placement is crucial for maintaining QoS in time-critical and drop-critical applications such as autonomous driving. Under proactive control, the RTT values were consistent, with low jitter and an average delay of only 0.1518 ms. In contrast, the reactive controller incurred longer delays and more variation due to the installation of real-time flow-rules, with RTTs ranging from 0.052 to 6.627 ms and an average of 2.0446 ms. This behavior demonstrates that proactive control is essential for maintaining QoS for real-time vehicular applications. These findings reinforce the value of pre-installing flow rules for time-critical and drop-critical traffic to ensure predictable and efficient network performance and to maintain QoS in SDIoV traffic handling.

Figure 11 shows the RTT associated with routing as traffic increases. The proposed work exhibits a marginal increase in routing delay compared to benchmark schemes, specifically FIFO and LRU schemes. The proposed method achieves a reduction in RTT of 32% compared to the FIFO scheme and 36% in comparison to the LRU scheme. The proposed work involves the controller classifying flows according to their priority.

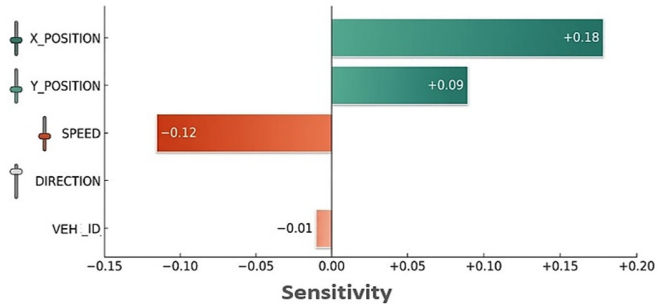

Fig. 8. Sensitivity analysis of the model.

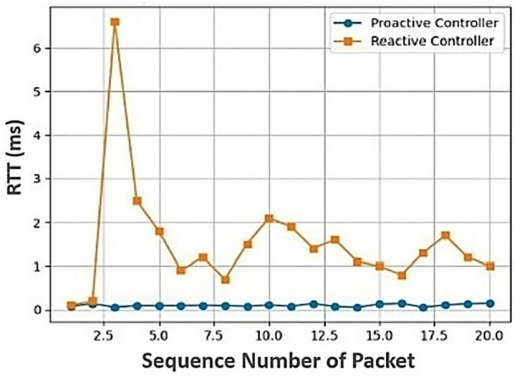

Fig. 9. RTT comparison for time-critical traffic.

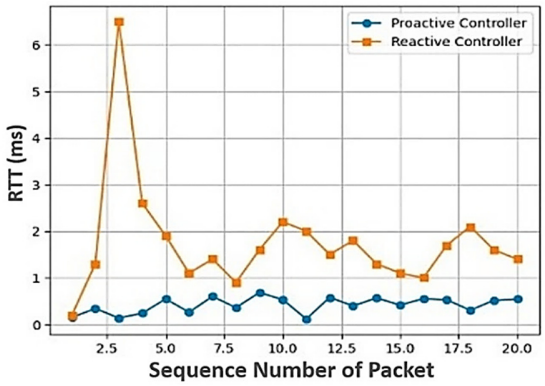

Fig. 10. RTT comparison for drop-critical traffic.

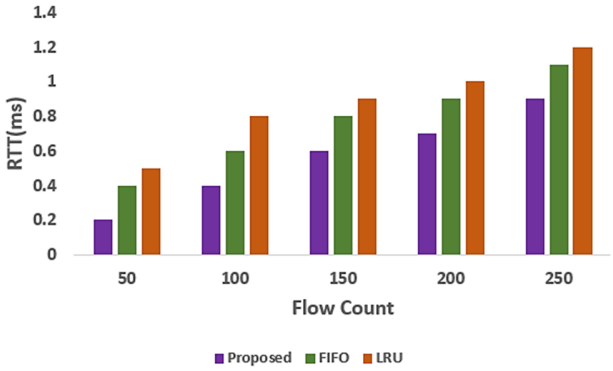

Fig. 11. RTT summary for time-critical traffic.

Figure 12 illustrates packet drops across a range of flows. The proposed method demonstrated a reduction in packet drops compared to the FIFO and LRU schemes. FIFO exhibited the lowest performance, with a significant percentage of packet drops occurring as the flow count increased, contributing to link congestion. This eventually results in additional packet drops within the network. The percentage of packet drop in the LRU scheme is lower than in the FIFO scheme because data is transmitted through multiple paths to account for network congestion. This results in a reduced packet drop percentile. In the proposed method, traffic is allocated by the QoS-Aware Queue Handler across multiple paths. This approach results in a lower packet drop rate and demonstrates superior performance compared to the other schemes.

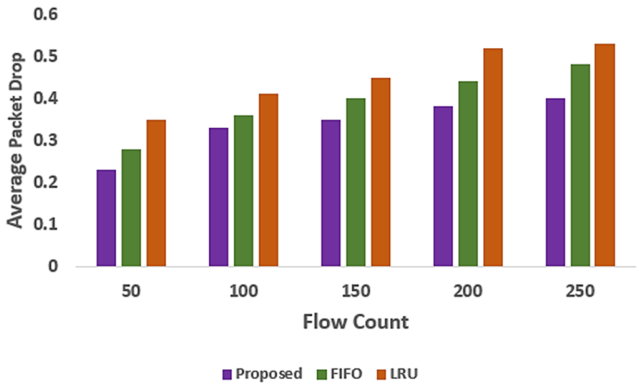

Fig. 12. RTT summary for drop-critical traffic.

Figure 13 shows the average jitter variation during packet flow with respect to the varying vehicle count range for the proposed scheme compared to FIFO and LRU. For maximum node count variation, the proposed scheme, being proactive, achieved less jitter by 25 ms, whereas FIFO and LRU had comparatively higher jitter value. Thus, with the proactive prediction of the AP and the QoS-Aware Queue Management, the packet drop probability is minimized in the network, which in turn reduces time delay variations.

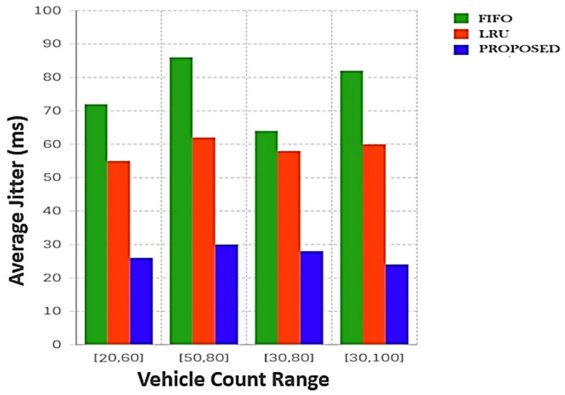

Fig. 13. Analysis of jitter variation with respect to node count variations.

IV. CONCLUSION

This paper presented an LSTM-based QoS-aware proactive flow-rule placement framework for SDIoV to address the limitations of conventional reactive SDN control in highly dynamic vehicular environments. By combining next AP prediction with DSCP-based traffic prioritization, the proposed framework enables proactive rule installation and differentiated handling of time-critical, drop-critical, and best-effort traffic. The results showed that the LSTM model had a validation accuracy of 80.88%. The proactive scheme improved RTT performance and reduced delay compared to reactive methods such as FIFO and LRU. These results demonstrate that integrating mobility prediction, QoS-aware traffic management, and proactive SDN control can significantly enhance vehicular network performance. Although the proposed framework achieved promising results, it was evaluated in a relatively small-scale setting that may not fully capture the challenges of dense IoV environments. The gap between training and validation accuracy also indicates the scope for improving model generalization. Future work will focus on large-scale evaluation, controller and flow-table scalability analysis, and the integration of advanced methods such as Gated Recurrent Unit (GRU), Transformer, and reinforcement learning to further enhance robustness and practical applicability.

DECLARATION OF COMPETING INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ACKNOWLEDGMENT

The authors would like to thank their institutions for providing the necessary facilities to carry out this research.

DATA AVAILABILITY

The data for this research were downloaded from the Caltrans Performance Measurement System (PeMS). The dataset used in this study is publicly available on [17].

REFERENCES

[1] P. Kamboj, S. Pal, S. Bera, and S. Misra, "QoS-Aware Multipath Routing in Software-Defined Networks," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 2, pp. 723–732, Mar. 2023, https://doi.org/10.1109/TNSE.2022.3219417.

[2] H. Ye, L. Liang, G. Ye Li, J. Kim, L. Lu, and M. Wu, "Machine Learning for Vehicular Networks: Recent Advances and Application Examples," *IEEE Vehicular Technology Magazine*, vol. 13, no. 2, pp. 94–101, June 2018, https://doi.org/10.1109/MVT.2018.2811185.

[3] A. S. Kumar, L. Zhao, and X. Fernando, "Multi-Agent Deep Reinforcement Learning-Empowered Channel Allocation in Vehicular Networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 1726–1736, Oct. 2022, https://doi.org/10.1109/TVT.2021.3134272.

[4] N. Cardona, E. Coronado, S. Latré, R. Riggio, and J. M. Marquez-Barja, "Software-Defined Vehicular Networking: Opportunities and Challenges," *IEEE Access*, vol. 8, pp. 219971–219995, 2020, https://doi.org/10.1109/ACCESS.2020.3042717.

[5] N. Saha, S. BERA, and S. Misra, "Sway: Traffic-Aware QoS Routing in Software-Defined IoT," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 1, pp. 390–401, Jan. 2021, https://doi.org/10.1109/TETC.2018.2847296.

[6] G. Huang, I. Ullah, H. Huang, and K. T. Kim, "Predictive mobility and cost-aware flow placement in SDN-based IoT networks: a Q-learning approach," *Journal of Cloud Computing*, vol. 13, no. 1, Jan. 2024, Art. no. 26, https://doi.org/10.1186/s13677-024-00589-w.

[7] B. E. Carpenter and K. Nichols, "Differentiated services in the Internet," *Proceedings of the IEEE*, vol. 90, no. 9, pp. 1479–1494, Sept. 2002, https://doi.org/10.1109/JPROC.2002.802000.

[8] Y. Kong *et al.*, "Unlocking the Power of LSTM for Long Term Time Series Forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 11, pp. 11968–11976, Apr. 2025, https://doi.org/10.1609/aaai.v39i11.33303.

[9] A. S. Kumar, L. Zhao, and X. Fernando, "Asynchronous Federated Based Vehicular Edge Computation Offloading," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 12, pp. 19350–19360, Dec. 2024, https://doi.org/10.1109/TVT.2024.3444784.

[10] H. Al-Maliki, H. A. A. AL-Asadi, Z. A. Abduljabbar, and V. O. Nyangaresi, "Reliable Vehicular Ad Hoc Networks for Intelligent Transportation Systems based on the Snake Optimization Algorithm," *Engineering, Technology & Applied Science Research*, vol. 14, no. 6, pp. 18631–18639, Dec. 2024, https://doi.org/10.48084/etasr.8851.

[11] J. Jin, J. Gubbi, T. Luo, and M. Palaniswami, "Network architecture and QoS issues in the internet of things for a smart city," in *2012 International Symposium on Communications and Information Technologies (ISCIT)*, Oct. 2012, pp. 956–961, https://doi.org/10.1109/ISCIT.2012.6381043.

[12] N. McKeown *et al.*, "OpenFlow: enabling innovation in campus networks," *SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, Nov. 2008, https://doi.org/10.1145/1355734.1355746.

[13] "Ryu SDN Framework." https://ryu-sdn.org/.

[14] "Mininet: An Instant Virtual Network on Your Laptop (or Other PC)." https://mininet.org/.

[15] "TensorFlow." https://www.tensorflow.org/.

[16] "Caltrans PeMS," *California Department of Transportation*. https://pems.dot.ca.gov/.

[17] "sowmyap01/lstm." Mar. 23, 2026, [Online]. Available: https://github.com/sowmyap01/lstm.