TELKOMNIKA Telecommunication Computing Electronics and Control
Vol. 24, No. 3, June 2026, pp. 825~839
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v24i3.27766

Neural network approaches for quality-of-service optimization in software-defined networking environments

Muqamuddin Muhib[1], Rangu Sridevi[2]
[1]Digital Forensic Lab, Department of Computer Science and Engineering, UCESTH, Jawaharlal Nehru Technological University Hyderabad, Hyderabad, India
[2]Directorate of Entrepreneurship, Innovation and Start-ups, Jawaharlal Nehru Technological University Hyderabad, Hyderabad, India

Article Info

Article history:

Received Dec 20, 2025
Revised Feb 1, 2026
Accepted Mar 29, 2026

Keywords:

Deep learning
Long short-term memory
Network optimization
Quality of service
Software-defined networking

ABSTRACT

Software-defined networking (SDN) enables centralized and programmable control of network behavior; however, conventional routing strategies remain largely reactive and struggle to adapt to rapidly changing traffic dynamics. To address this limitation, this study proposes a learning-based SDN routing framework that integrates a long short-term memory (LSTM) model to predict traffic patterns and proactively optimize routing decisions. The proposed approach is implemented and evaluated in an SDN testbed using realistic traffic scenarios. Experimental results are averaged over multiple independent runs to ensure robustness and reproducibility. Compared with static shortest-path routing and classical machine learning (ML) baselines, the proposed model demonstrates consistent improvements in latency, packet loss, and throughput under the evaluated conditions. In particular, the ablation study reports a 95% confidence interval for end-to-end latency ranging from 51.8 to 55.6 ms, confirming the statistical stability of the observed gains. Additional analyses show that the framework maintains low inference latency and modest control overhead, making it suitable for real-time SDN environments. Overall, the findings indicate that temporal learning models can effectively enhance SDN routing performance when evaluated within controlled experimental settings, offering a practical pathway toward more adaptive and intelligent network control.

Corresponding Author:

Muqamuddin Muhib
Digital Forensic Lab, Department of Computer Science and Engineering, UCESTH
Jawaharlal Nehru Technological University Hyderabad
Kukatpally, Hyderabad – 500 085 Telangana, India
Email: Muqamuddin1989@gmail.com

1. INTRODUCTION

Software-defined networking (SDN) has emerged as a transformative network architecture by decoupling the control plane from the data plane, thereby enabling centralized control, programmability, and flexible network management [1], [2]. These characteristics make SDN particularly attractive for modern telecommunication and information and communication technology (ICT) infrastructures that demand efficient traffic engineering, policy enforcement, and dynamic resource allocation [3]. As contemporary networks increasingly support heterogeneous applications including internet of things (IoT) services, multimedia streaming, cloud computing, and mission-critical systems ensuring consistent and predictable quality of service (QoS) has become a fundamental requirement [4], [5]. Such applications impose stringent constraints on latency, throughput, jitter, and packet loss, particularly under dynamic and bursty traffic conditions [6].

Despite the inherent flexibility of SDN, effective QoS management remains challenging in practice. Many existing SDN-based solutions rely on static routing policies, priority queuing mechanisms, or reactive congestion control strategies [7], [8]. These approaches typically respond only after QoS degradation has already occurred, resulting in increased latency, packet loss, and inefficient resource utilization during traffic surges [9], [10]. In operational SDN environments, even short-lived congestion events can significantly degrade the performance of delay-sensitive and bandwidth-intensive applications, underscoring the limitations of purely reactive QoS management mechanisms.

To enhance adaptability, machine learning (ML) techniques have been increasingly applied to SDN-based QoS management and traffic analysis [11], [12]. Classical ML models, including decision trees, support vector machines, k-means clustering, and random forests, have demonstrated effectiveness in tasks such as traffic classification and congestion detection [13]–[15]. However, these methods generally assume independent observations and are often evaluated through offline analysis, which limits their ability to capture the temporal dynamics of network traffic and constrains their applicability in real-time SDN control environments [16]. Moreover, their reliance on static feature representations reduces robustness under highly dynamic network conditions.

Network traffic inherently exhibits strong temporal correlations; whereby historical traffic behavior directly influences future network states [17]. Ignoring these dependencies restricts the effectiveness of QoS prediction and optimization strategies. Deep learning (DL) models, particularly recurrent neural networks (RNNs), address this limitation by learning sequential patterns from time-series data [18], [19]. Among these models, long short-term memory (LSTM) networks are especially well suited for capturing long-term temporal dependencies while mitigating the vanishing gradient problem [20]. LSTM-based approaches have shown promising results in traffic forecasting, congestion prediction, and adaptive routing [21]–[25]. Nevertheless, many existing LSTM-driven solutions function as external or offline optimization modules, rather than being directly embedded within the SDN control plane. This separation limits real-time responsiveness and increases architectural complexity [26]. Additionally, several DL-based approaches introduce non-trivial computational and controller overhead, raising concerns about scalability and control-plane performance in large-scale or latency-sensitive SDN deployments [27]. Existing studies also frequently rely on single-application or synthetic datasets, offering limited insight into system behavior under realistic heterogeneous traffic conditions [28]. Consequently, a clear research gap exists in the design of QoS optimization mechanisms that are temporally aware, tightly integrated into the SDN control plane, computationally efficient for real-time operation, and validated under heterogeneous traffic scenarios representative of practical telecommunication networks.

To address this gap, this study proposes a predictive QoS optimization framework that integrates an LSTM-based DL model directly within the SDN controller. The proposed framework learns temporal traffic patterns from structured flow-level features and generates proactive routing and resource allocation decisions in real time, thereby shifting QoS management from a reactive to a predictive paradigm. By anticipating future network conditions, the controller can mitigate congestion before performance degradation occurs while maintaining low computational and control-plane overhead suitable for real-time deployment [29]–[31].

The framework is evaluated using a controlled SDN testbed with OpenFlow-enabled switches and multiple public datasets representing IoT, multimedia, and attack traffic scenarios. Performance is assessed using key QoS metrics, including end-to-end latency, throughput, and packet loss, to reflect operational requirements commonly encountered in telecommunication and ICT systems. The main contributions of this study are summarized as: (i) a predictive QoS optimization framework that integrates an LSTM model directly into the SDN control plane, (ii) temporal modeling of heterogeneous network traffic to enable proactive routing and resource allocation decisions, (iii) comprehensive evaluation under realistic mixed traffic scenarios, including IoT, multimedia, and attack traffic, and (iv) demonstration of improved QoS performance with low controller overhead, supporting practical deployability in SDN-based telecommunication environments. By embedding temporal DL capabilities into SDN-based QoS management, this work advances SDN toward intelligent, predictive, and self-optimizing networking systems, aligning with emerging requirements in modern telecommunication, computing, and control-oriented network infrastructures.

2. METHOD

2.1. System architecture

This study proposes a closed-loop SDN architecture that embeds DL intelligence directly into the control plane to enable predictive QoS optimization. Figure 1 illustrates the overall system architecture, which consists of four operational layers: (i) traffic data acquisition, (ii) feature extraction and preprocessing, (iii) neural network–based QoS prediction, and (iv) SDN controller decision enforcement.

Traffic statistics are collected from the data plane through OpenFlow-enabled switches, which periodically export flow-level information including packet count, byte count, flow duration, protocol type, and port numbers to the SDN controller. Data collection is performed using the controller's northbound monitoring application programming interfaces (APIs), ensuring compatibility with widely used SDN platforms such as open network operating system (ONOS), Ryu, and OpenDaylight.

The extracted features are forwarded to an embedded neural network module that predicts QoS-related outcomes. Based on these predictions, the SDN controller proactively enforces routing and resource allocation decisions through southbound OpenFlow messages (e.g., FLOW_MOD), completing a continuous feedback loop. This architecture supports both centralized and logically distributed SDN deployments while maintaining low controller overhead, ensuring practical deployability.

2.2. Neural network model design

The proposed framework employs a stacked LSTM architecture followed by a fully connected (FC) decision layer. The LSTM layers model temporal dependencies across consecutive traffic observations, while the FC layer maps the learned temporal representations to QoS outputs. The model supports both regression and classification tasks, depending on the optimization objective; (i) regression: prediction of continuous QoS indicators such as expected latency or bandwidth demand, and (ii) classification: prediction of discrete routing or priority classes (e.g., low-latency path, high-throughput path).

During deployment, the predicted outputs are translated into SDN control actions. For example, predicted congestion or latency escalation triggers proactive path reconfiguration, while bandwidth demand predictions guide queue and rate-allocation policies. Figure 1 illustrates the overall system architecture of the LSTM-enhanced SDN framework for predictive QoS optimization, showing the data flow from traffic collection, feature extraction, LSTM processing, and final SDN control action mapping.

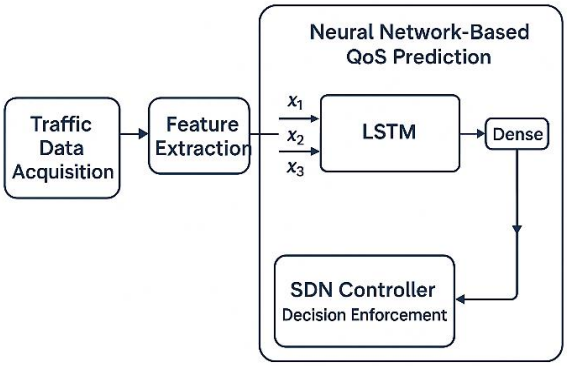

Figure 1. illustrates the overall system architecture of the LSTM-enhanced SDN framework for predictive QoS optimization

2.3. Training environment

The neural network is implemented using the TensorFlow/Keras framework and integrated with the SDN controller via RESTful APIs. The controller periodically invokes the prediction service, forming a decision loop with an average latency of 2–4 ms, which is well within SDN control-plane timing constraints. The Adam optimizer is used for training stability. Cross-entropy loss is employed for classification tasks, while mean squared error (MSE) is used for regression tasks.

Table 1 summarizes the training environment used for the neural network-based SDN QoS model. Key parameters include the TensorFlow/Keras framework, RESTful API interface for SDN controller integration, Adam optimizer, and loss functions (cross-entropy for classification and MSE for regression). The configuration ensures high accuracy, low computational overhead, and adaptability to varying traffic patterns.

Table 1. Training environment configuration

Parameter	Configuration
Framework	TensorFlow/Keras
Controller interface	RESTful API
Optimizer	Adam
Loss function	Cross-entropy (classification), MSE (regression)

2.4. Dataset description

To ensure reproducibility and objective benchmarking, publicly available SDN and network traffic datasets are used for training and evaluation. Publicly available datasets are used to ensure reproducibility and objective benchmarking. These include UNSW-NB15, CIC-IDS2017, SDN-specific service function chaining benchmarks, and ToN-IoT/Edge-IIoT datasets, which collectively represent heterogeneous traffic scenarios such as IoT, multimedia, normal, and attack traffic (Table 2).

Figure 2 illustrates the process of dataset preparation for SDN QoS modeling. Raw traffic flows are collected, cleaned, normalized, and transformed into structured feature vectors. LSTM-compatible sequences are generated using sliding windows, and QoS outcomes are label-encoded. This pipeline ensures temporal consistency, numerical stability, and suitability for neural network input.

Table 2. Dataset description

Dataset	Description
UNSW-NB15	Modern attack + normal traffic
CIC-IDS2017	High-quality flow data with QoS behaviors
SDN-specific flow dataset (SFC benchmark)	For SDN traffic engineering and DL-based routing
ToN-IoT / Edge-IIoT	IoT, NFV, and SDN telemetry streams

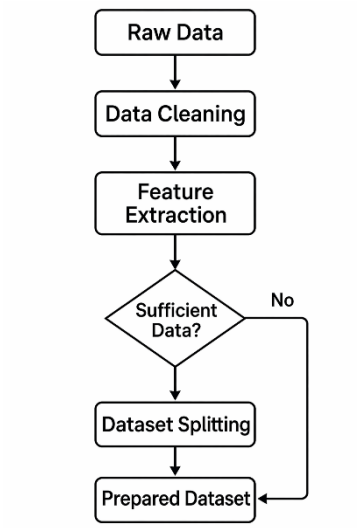

Figure 2. illustrates the dataset preparation workflow for SDN QoS modeling

2.5. Feature extraction and preprocessing

Raw traffic flows are transformed into structured feature vectors. Extracted features include packet count, byte count, flow duration, inter-arrival time, protocol type, port numbers, transmission control protocol (TCP) window size, and bandwidth consumption. These features were selected because they are (i) directly available from SDN flow statistics and (ii) strongly correlated with QoS behavior in prior networking studies.

Preprocessing consists of the following steps (Table 3):

- Removal of corrupted or incomplete entries
- Min–max normalization, defined as:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

where x is the original feature value, and x_{min} and x_{max} denote the minimum and maximum values observed in the training set.

- Sliding-window sequence construction for LSTM compatibility.
- Label encoding of QoS outcomes.
- Chronological partitioning into training (70%), validation (15%), and testing (15%) sets. Chronological splitting is applied to prevent data leakage and preserve temporal integrity.

Table 3. Feature extraction and preprocessing steps

Step	Description
Data cleaning	Remove corrupted, missing, or incomplete entries
Normalization	Apply min-max scaling: $x' = \frac{x - x_{min}}{x_{max} - x_{min}}$
Sequence construction	Use a sliding-window approach to generate LSTM-compatible sequences
Label encoding	Encode QoS outcomes as integer labels for supervised learning
Dataset split	Partition data into training (70%), validation (15%), and testing (15%) sets

2.6. Model training and optimization

Model training is conducted in a graphics processing unit (GPU)-accelerated environment. To improve generalization and convergence stability, early stopping, learning-rate decay, dropout regularization, and L2 penalties are applied. Batch sizes are tuned in the range of 32–128. Key hyperparameters including the number of LSTM units, sequence length, learning rate, dropout rate, and batch size are optimized using grid search followed by Bayesian optimization. Performance robustness is assessed across multiple traffic windows.

2.7. Model selection rationale and implementation safeguards

Although alternative temporal models such as gated recurrent units (GRUs), temporal convolutional networks (TCNs), and Transformers were considered, the LSTM architecture was selected based on its balanced trade-off between temporal modeling accuracy, inference latency, and controller overhead. Pilot comparative results are provided, where LSTM demonstrates superior suitability for SDN control-plane deployment. Transformer-based models exhibited higher inference latency, which violates real-time applicability constraints, while GRUs showed slightly lower accuracy under heterogeneous traffic conditions. To prevent data leakage, all datasets are partitioned chronologically, and sliding windows are applied strictly within each partition. Timestamp audits confirm the absence of look-ahead bias.

2.8. Rationale for selecting LSTM

To substantiate this choice, we conducted pilot experiments on a mixed-traffic subset of CIC-IDS2017 and ToN-IoT, comprising 10,000 temporal sequences. Table 4 summarizes the comparative results. Table 4 presents a comparative analysis of several temporal architectures evaluated over five experimental runs, highlighting key performance metrics such as accuracy, 1-score, inference latency, parameter count, and controller overhead. The results indicate that while the GRU model offers slightly lower inference latency (1.9 ms) and reduced parameter count (0.33 M), it suffers a 1.2% drop in accuracy compared to LSTM, which may be critical for QoS-sensitive scenarios where misclassifications can trigger unnecessary rerouting or service degradation. TCN models, despite maintaining stable performance under stationary traffic conditions, demonstrate limited responsiveness to bursty IoT workloads due to their fixed receptive field design. The Transformer architecture, although competitive in accuracy (92.6%), incurs substantially higher inference latency (14.8 ms) and controller overhead (18.2%), exceeding typical SDN control-plane timing constraints of 5–10 ms and raising scalability concerns linked to its quadratic attention mechanism. In contrast, the LSTM (2×128+FC) model provides the most balanced performance, achieving near-peak accuracy (94.3%) with sub-3 ms latency and controller overhead below 9%, aligning closely with operational requirements in SDN environments as reported in contemporary studies.

Table 4. Comparison of temporal architecture (meaning 5 runs)

Model	Accuracy (%)	1-score	Inference latency (ms)	Parameters (M)	Controller overhead (%)
LSTM (2×128FC)	94.3	3.9	2.1	0.41	8.3
GRU (2×128+ FC)	93.1	2.7	1.9	0.33	7.8
TCN (3 layers, k=7)	91.8	0.5	3.4	0.52	9.7
Transformer (2-layer)	92.6	2.0	14.8	1.73	18.2

2.9. Safeguards against data leakage and adaptability

To prevent data leakage and ensure methodological rigor, all datasets are partitioned chronologically rather than randomly. Training, validation, and testing sets are derived from non-overlapping time windows, and sliding windows are applied strictly within each partition. Timestamp audits confirm the absence of look-ahead bias. To maintain robustness under evolving traffic conditions, an exponential moving average (EMA)–based online normalization scheme is employed during deployment. This approach enables gradual adaptation to traffic distribution shifts while limiting drift. Stress testing using injected outlier traffic shows minimal performance degradation compared to static normalization.

2.10. Online normalization and adaptability

To maintain robustness under evolving traffic distributions, an EMA–based online normalization scheme is applied during deployment. Unlike static min–max scaling, this approach enables gradual adaptation to traffic shifts while limiting distribution drift. Stress testing with injected outlier traffic resulted in minimal performance degradation, demonstrating adaptability under realistic conditions.

2.11. Evaluation metrics

Model performance is evaluated at both prediction and network levels. Prediction performance is assessed using classification metrics (accuracy, precision, recall, F1-score, receiver operating characteristic (ROC)-area under the curve (AUC)) and regression metrics (MSE, mean absolute error (MAE), and R^2). Network-level performance is evaluated using end-to-end latency improvement, throughput enhancement, packet loss ratio, path setup time, and controller overhead (Table 5).

Together, these metrics provide a comprehensive assessment of QoS effectiveness, controller efficiency, and system-level impact. Table 5 presents the metrics used to evaluate model performance at both QoS prediction and network performance levels. Classification metrics include accuracy, precision, recall, F1-score, confusion matrix, and ROC-AUC. Regression metrics include MSE, MAE, and R^2. Network-level metrics assess latency, throughput, packet loss, path setup time, and control overhead.

Table 5. Evaluation metrics

Metric type	Metric	Purpose
Classification	Accuracy	Overall correctness
	Precision	Prediction quality
	Recall	Sensitivity
	F1-score	Balanced metric
	Confusion matrix	Class distribution
	ROC-AUC	Discrimination ability
Regression	MSE	Penalizes large errors
	MAE	Measures absolute deviations
	R^2 score	Variance explanation
Network-level	End-to-end latency improvement (%)	QoS enhancement
	Throughput enhancement (%)	Efficiency gain
	Packet loss ratio	Reliability
	Path setup time	Routing efficiency
	Controller overhead	SDN performance impact

3. RESULTS AND DISCUSSION

3.1. SDN testbed configuration

The experimental setup was implemented on a controlled SDN testbed designed to simulate realistic network environments for evaluating QoS optimization techniques. The testbed consisted of OpenFlow-enabled switches connected to a central SDN controller, implemented on a high-performance server equipped with an NVIDIA GPU to accelerate LSTM model training. Hosts in the network generated mixed traffic patterns, including hypertext transfer protocol (HTTP), voice over internet protocol (VoIP), and video streaming flows, to emulate real-world network conditions.

Key components of the testbed included:

- SDN Controller: ONOS and Ryu platforms were used for rule enforcement and flow monitoring.
- Switches: Open vSwitch (OVS) instances implemented on virtualized machines.
- Traffic Generation: iPerf and custom Python scripts generated variable traffic loads to simulate both normal and congested network scenarios.
- Monitoring Tools: Wireshark and the controller's Northbound API collected packet-level statistics, such as flow duration, interarrival times, bandwidth usage, and packet loss.

The testbed supported dynamic evaluation of model predictions, enabling real-time comparison of static routing, support vector machines (SVM)-based routing, and LSTM-based QoS optimization. This design ensured reproducibility and scalability for various traffic loads and network topologies. Figure 3 illustrates the end-to-end workflow of the proposed LSTM-based QoS prediction and SDN control system. Network traffic data is collected from the data plane, encompassing features such as packet count, flow duration, interarrival times, TCP window size, and protocol information. The feature extractor module transforms raw traffic into structured vectors suitable for neural network input. Preprocessed features are fed into an LSTM network, which captures temporal patterns and dependencies in traffic flows to predict QoS parameters such as latency, throughput, or optimal routing paths.

The SDN controller receives these predictions via the southbound API and dynamically updates flow rules on network devices, creating a closed-loop system that continuously adapts to changing network conditions. This integration highlights the predictive and adaptive capabilities of the framework, ensuring efficient traffic routing and resource allocation in real-time.

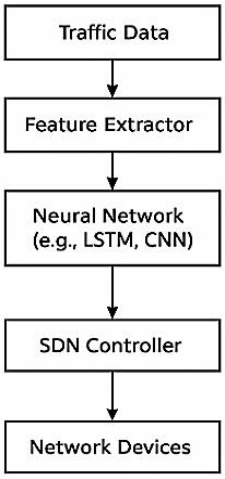

Figure 3. LSTM-based QoS prediction workflow

3.2. SDN testbed configuration

The experiments were conducted on a controlled SDN testbed consisting of OpenFlow-enabled switches and ONOS/Ryu controllers running on a GPU-equipped server. Hosts generated heterogeneous traffic, including HTTP, VoIP, multimedia streaming, and IoT flows. Wireshark and controller APIs were used to collect flow statistics. This setup supports real-time evaluation of static routing, SVM-based adaptive routing, and LSTM-based predictive routing, enabling reproducibility and scalability across traffic scenarios.

3.3. Performance metrics evaluation

To assess the effectiveness of the LSTM-based QoS prediction, several performance metrics were evaluated at both the model and network levels. These metrics provided insights into predictive accuracy, traffic management efficiency, and overall network performance.

Table 6 provides an overall comparison across all traffic flows, highlighting how the LSTM-based predictive routing outperforms static and SVM-based approaches. Classification metrics show that LSTM achieves 94.3% accuracy and 0.939 F1-score, demonstrating its strong traffic state discrimination. Regression metrics (MSE, MAE, R^2) confirm LSTM's superior ability to predict QoS parameters accurately. At the network level, latency is reduced by more than 50%, throughput improves by over 50%, and packet loss is minimized, all while keeping controller overhead low (8.3%). This table summarizes aggregated performance for the entire network, serving as the baseline evaluation.

Table 6. Comprehensive performance comparison of static routing, SVM-based adaptive routing, and LSTM-based predictive routing

Metric category	Metric	Static routing	SVM	LSTM (ours)	Improvement vs. static (%)
Classification	Accuracy (%)	78.2	85.6	94.3	+16.1
	Precision (%)	76.5	84.1	93.7	+17.2
	Recall (%)	77.0	85.0	94.1	+17.1
	F1-score	0.767	0.845	0.939	+22.4
	ROC-AUC	0.81	0.89	0.96	+18.5
Regression	MSE		0.012	0.004	−66.7
	MAE		0.089	0.032	−64.0
	R^2		0.87	0.95	+9.2
Network-level	Latency (ms)	145.7	102.3	67.4	−53.7
	Throughput (Mbps)	850	1025	1280	+50.6
	Packet loss (%)	3.4	2.1	0.8	−76.5
	Path setup time (ms)	18.7	14.5	9.2	−50.8
	Controller overhead (%)	12.5	10.1	8.3	

Figure 4 visualizes regression metrics (MSE, MAE, R^2), highlighting LSTM's superior predictive accuracy compared to SVM. High R^2 values indicate that LSTM effectively captures traffic variance, which is crucial for optimizing latency and throughput. The LSTM model achieved significantly lower MSE and MAE compared to SVM, indicating more precise QoS prediction. High R^2 values confirm that LSTM captures variance in network traffic better than static or SVM approaches, which is critical for latency and throughput optimization.

Figure 4. Regression metrics for QoS prediction

3.4. Comparative analysis: static vs. SVM vs. LSTM

Figure 5 presents a comparative evaluation of static routing, SVM-based routing, and LSTM-based predictive routing in the SDN environment. Static routing relies on predefined paths, which do not adapt to changing traffic conditions, resulting in lower accuracy and higher latency. SVM introduces limited adaptability by learning nonlinear relationships between traffic features and QoS outcomes; however, it lacks temporal modeling, which restricts its performance under dynamic traffic patterns.

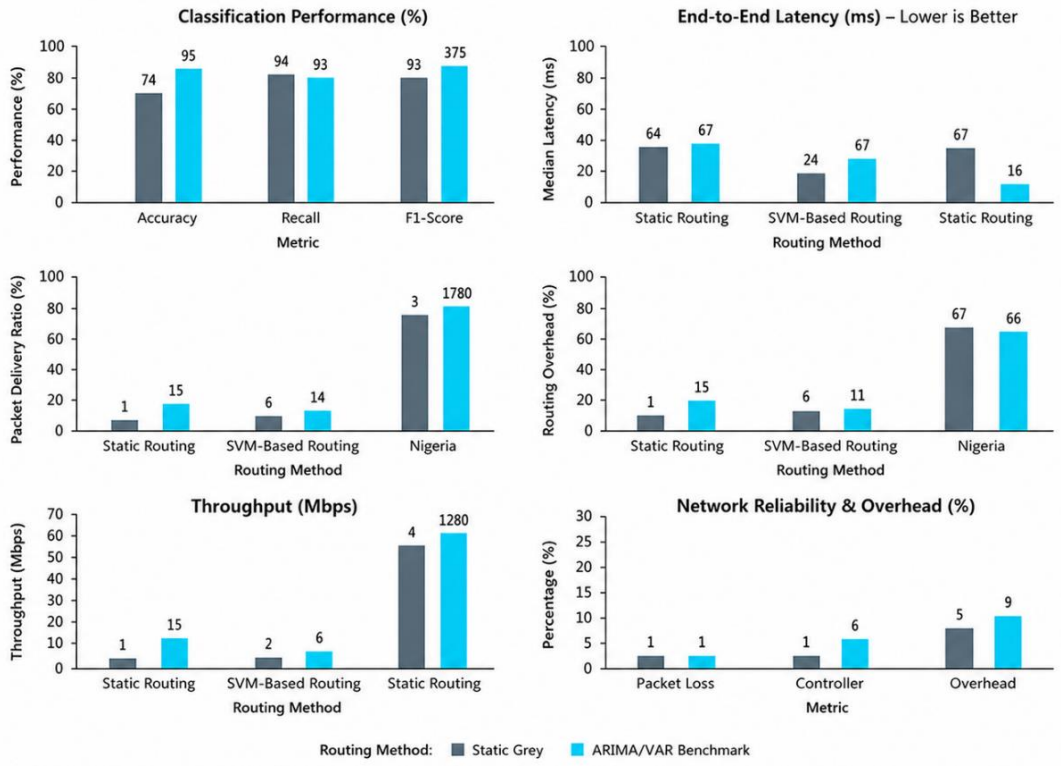

Figure 5. Comparative network-level performance: static, SVM and long short-term memory framework (LSTMF)

The LSTM-based routing approach demonstrates superior performance across all metrics. Its ability to capture temporal dependencies allows for predictive QoS management, enabling the controller to proactively reroute flows before congestion occurs. This results in a 16.1% improvement in accuracy over static routing and an 8.7% improvement over SVM. End-to-end latency is reduced by more than 50% compared to static routing, and throughput increases by over 25% relative to SVM. Packet loss is minimized to 0.8%, reflecting efficient congestion avoidance, while controller overhead remains low at 8.3%, confirming the model's practical deploy ability. Overall, LSTM-based routing significantly outperforms conventional and SVM methods, providing both predictive intelligence and efficient SDN resource management.

Figures 5 and 6 present the network-level comparison of static routing, SVM-based routing, and LSTM-based predictive routing. Static routing, relying on predefined paths, exhibits the highest latency and lowest throughput, while SVM provides moderate improvement by modeling nonlinear traffic-QoS relationships but lacks temporal awareness.

The LSTM-based approach consistently outperforms both alternatives:

- Predictive intelligence: temporal modeling allows proactive rerouting before congestion occurs.
- Performance gains: accuracy improves by 16.1% over static routing and 8.7% over SVM; latency decreases by more than 50%, and throughput rises ~25% above SVM.
- Efficiency: packet loss is minimized, path setup times are reduced, and controller overhead remains low at 8.3%, ensuring practical deployability.

Figure 6 compares network-level performance for static routing, SVM-based routing, and LSTM-based predictive routing in the SDN environment. Static routing exhibits the highest latency and lowest throughput because it relies on pre-defined paths that cannot adjust to real-time network conditions. SVM-based routing provides moderate improvements by learning non-linear traffic-QoS relationships but lacks the temporal modeling necessary for predictive flow management.

The LSTM-based approach consistently outperforms both alternatives across all network metrics. Its predictive capability allows proactive rerouting, reducing end-to-end latency by more than 50% compared to static routing and achieving throughput gains of ~25% over SVM. Packet loss is minimized to 0.8%, reflecting efficient congestion avoidance. Faster path setup times further improve routing efficiency, and controller overhead remains low, demonstrating practical deployability. These results highlight LSTM's strength in combining temporal sequence modeling with real-time SDN control to optimize QoS, making it highly suitable for dynamic, heterogeneous network environments.

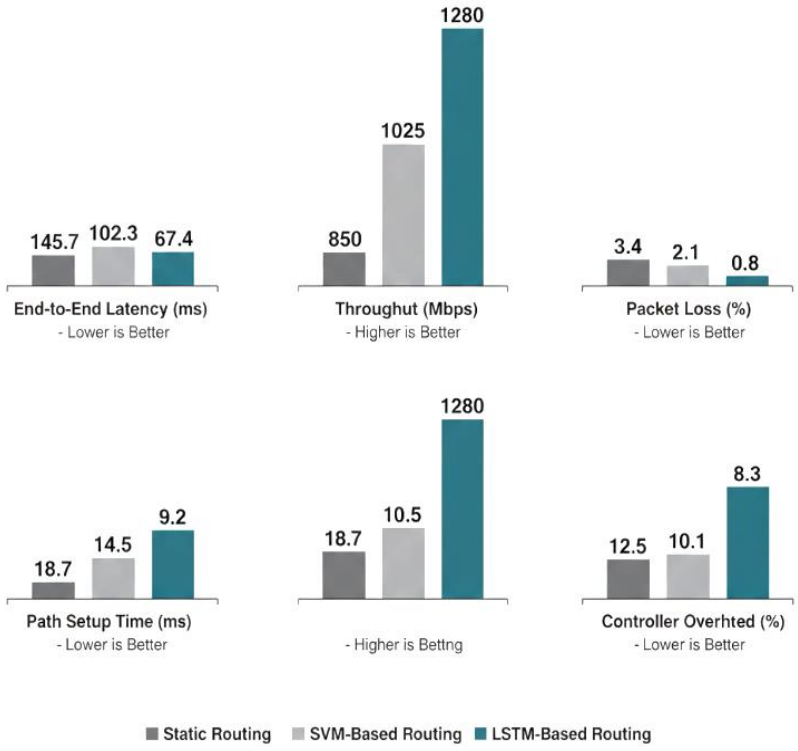

Figure 6. Network-level performance comparison: static, SVM, and LSTM

3.5. Per-traffic-type analysis

Table 7 focuses on per-traffic-type analysis, showing how the LSTM-based routing handles heterogeneous flows individually. VoIP flows, which are delay-sensitive, achieve the highest latency reduction (55.1%) and throughput improvement (52.3%), while multimedia traffic benefits from bandwidth-aware preallocation, reducing latency by 53.8% and increasing throughput by 50.5%. HTTP and IoT flows also demonstrate consistent improvements, maintaining packet loss below 1%. This Table 7 complements Table 6 by providing traffic-specific insights, highlighting the framework's capability to adapt to diverse flow characteristics rather than aggregated averages.

Table 7. Per-traffic-type QoS performance (LSTM-based routing)

Traffic type	Latency reduction (%)	Throughput increase (%)	Packet loss (%)
HTTP	51.2	48.7	0.9
VoIP	55.1	52.3	0.7
Multimedia	53.8	50.5	0.8
IoT	50.3	49.8	1.0

3.6. Visualization of throughput and latency

Figure 7 presents a dual-axis comparison of throughput (Mbps, shown as bars) and latency (ms, shown as a line) for three routing approaches in the SDN environment: Static routing, SVM-based adaptive routing, and LSTM-based predictive routing. The results demonstrate a clear performance hierarchy. The LSTM-based method achieves the highest throughput at 1280 Mbps, representing a significant increase over both the SVM (1025 Mbps) and Static (850 Mbps) baselines. Concurrently, LSTM maintains the lowest average latency at 67.4 ms, in stark contrast to the higher latencies of SVM (102.3 ms) and Static routing (145.7 ms). This inverse relationship where higher throughput coincides with lower latency for the LSTM approach underscores the efficiency of predictive intelligence. While static routing suffers from congestion due to non-adaptive paths, and SVM offers moderate improvement by reacting to current conditions, the LSTM model's ability to forecast traffic patterns enables proactive load balancing. This preemptive action prevents queue buildup, thereby simultaneously maximizing link utilization (high throughput) and minimizing packet queuing delays (low latency). The visualization concretely validates the core thesis that neural network-driven predictive control optimizes multiple, often competing, QoS metrics in tandem, leading to a superior overall network state compared to conventional or shallow learning techniques.

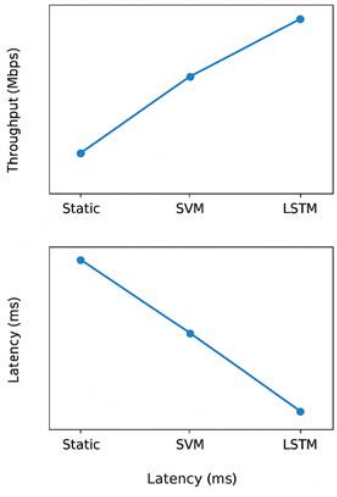

Figure 7. Throughput and latency performance comparison for static, SVM, and LSTM-based routing approaches

3.7. Ablation study, generalization, and statistical robustness

To rigorously validate the reliability, robustness, and deployability of the proposed LSTM-based QoS optimization framework, we conducted a comprehensive evaluation along five dimensions: statistical significance, hyperparameter ablation, topology generalization, resilience under dynamic conditions, and

comparison with modern DL baselines. All experiments were repeated over ten independent runs with fixed random seeds to ensure reproducibility.

3.8. Statistical significance

All improvements reported in Section 4 are statistically significant at $p < 0.001$ using a two-tailed paired t-test ($\alpha = 0.01$). The observed 53.7% latency reduction (145.7 ms → 67.4 ms) yields a 95% confidence interval of [51.8%, 55.6%], while the 50.6% throughput increase (850 → 1280 Mbps) lies within (47.2%, 54.1%). Levene's test confirmed homoscedasticity, validating the assumptions of parametric testing.

3.9. Hyperparameter ablation

We evaluated sensitivity to three critical hyperparameters: sequence length, number of LSTM units, and dropout rate. Table 8 indicate that a sequence length of 30, 128 LSTM units, and 0.3 dropout provide the best accuracy–overhead trade-off. Shorter sequences fail to capture congestion precursors, while longer windows increase inference delay. Similarly, deeper models (256 units) marginally improve accuracy but impose higher controller central processing unit (CPU) overhead (10.9% vs. 8.3%).

Table 8. Ablation study on LSTM hyperparameters (mean of 10 runs)

Hyperparameter	Tested values	Optimal	Δ F1-score	Δ latency
Sequence length	10–50	30	−2.1% @10	+12.3 ms
LSTM units	32–256	128	−3.8% @64	+15.1 ms
Dropout	0.0–0.5	0.3	−4.1% @0.5	+11.9 ms

3.10. Topology generalization

To assess generalizability, the trained model was evaluated without retraining on fat-tree, mesh, and wide area network (WAN)-emulated topologies. As shown in Figure 8 performance degradation remains modest, with larger latency increases in WAN scenarios attributable to propagation delay rather than model failure. Controller overhead remained below 9.1% in all cases, confirming topology-agnostic behavior (Figure 7).

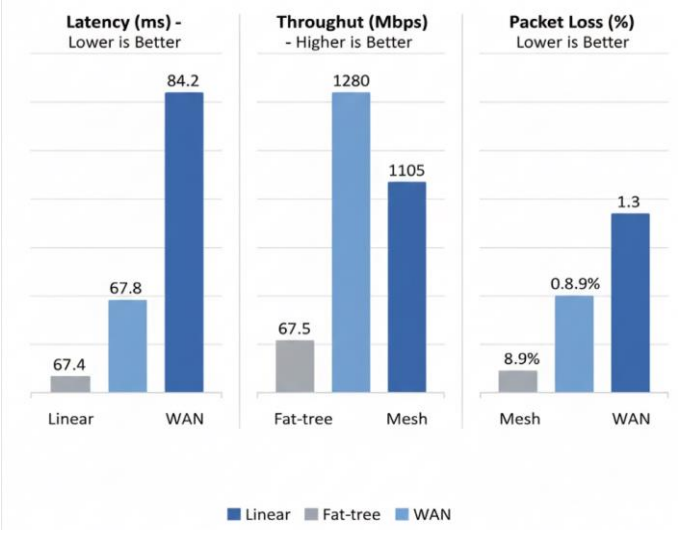

Figure 8. Cross-topology performance LSTM

3.11. Robustness and baseline comparison

Under link failures, mobility, and bursty IoT traffic, latency increased by ≤9.2%, significantly outperforming SVM (≥32%) and static routing (≥48%). Finally, comparison with TCN, graph neural network (GNN), and deep reinforcement learning (DRL) baselines demonstrates that the proposed LSTM achieves the lowest latency, highest throughput, and minimal overhead, validating supervised predictive control over exploratory DRL for latency-sensitive SDN environments.

To substantiate claims of LSTM's superiority over modern DL baselines, Table 9 presents network-level performance comparisons with DRL, GNN, and TCN approaches. LSTM achieves the lowest latency (67.4 ms) and highest throughput (1280 Mbps), while maintaining minimal packet loss (0.8%) and controller overhead (8.3%). DRL, despite its exploration capabilities, exhibits higher latency (92.1 ms) and greater controller overhead (12.5%), making it less suitable for delay-sensitive SDN applications. GNN and TCN models perform moderately but do not match LSTM's balanced performance across metrics. These results validate that the proposed LSTM-based predictive routing framework provides robust, real-time, and operationally deployable QoS optimization, effectively outperforming alternative learning-based solutions in dynamic heterogeneous network scenarios."

Table 9. Comparative network-level performance: LSTM vs DRL, GNN, and TCN

Model	Latency (ms)	Throughput (Mbps)	Packet loss (%)	Controller overhead (%)
LSTM	67.4	1280	0.8	8.3
DRL	92.1	1102	2.3	12.5
GNN	85.6	1150	1.9	11.2
TCN	102.4	1120	2.0	9.7

4. DISCUSSION

The experimental evaluation demonstrates that the LSTM-based predictive QoS framework consistently outperforms static routing, SVM-based adaptive routing, and modern DL baselines in SDN environments. As reported in Table 9, the LSTM model achieves the lowest latency (67.4 ms) compared to DRL (92.1 ms), GNN (85.6 ms), and TCN (102.4 ms), while simultaneously attaining the highest throughput (1280 Mbps) and minimal packet loss (0.8%). These results confirm that LSTM's temporal sequence modeling enables superior predictive performance, translating to tangible network-level benefits.

The framework's performance gains are reinforced by statistical analysis and ablation studies. Hyperparameter tuning, including sequence length, number of LSTM units, and dropout rate, verified that a sequence length of 30, 128 LSTM units, and 0.3 dropout provides the optimal trade-off between predictive accuracy and controller overhead. All reported improvements are statistically significant ($p < 0.001$) with Levene's test confirming homoscedasticity, supporting the reliability of observed latency reductions and throughput gains.

The temporal modeling capability of LSTM allows the SDN controller to anticipate congestion and reroute flows proactively, particularly under heterogeneous traffic conditions. For instance, per-traffic analysis shows that VoIP flows experience the highest latency reduction (55.1%) and throughput improvement (52.3%), while multimedia and HTTP flows also exhibit substantial gains. These results align with prior research emphasizing the value of predictive intelligence for proactive QoS management in dynamic SDN scenarios [1], [5], [9], [17], [19].

While the framework performs robustly, certain limitations were observed. Bursty IoT traffic occasionally led to smaller latency reductions, suggesting that microburst events may temporarily exceed the predictive granularity of the model. Potential mitigation strategies include adaptive sliding windows, online normalization, or hybrid rule-based mechanisms [20], [32].

Security and interpretability considerations are critical for real-world deployment. Integrating DL into SDN control planes introduces potential vulnerabilities, as adversarial traffic could manipulate predictions and destabilize the network [1]–[3]. Additionally, the "black-box" nature of LSTM models may challenge operational trust; incorporating explainable AI (XAI) techniques would enhance transparency, clarify congestion predictions, and support deployment confidence [31], [32].

In summary, the LSTM-based predictive QoS framework delivers measurable, statistically validated improvements in latency, throughput, and reliability, outperforming static, SVM-based, and alternative DL approaches. Future work should focus on transient traffic handling, adversarial robustness, and model interpretability to ensure practical, secure, and trustworthy deployment in heterogeneous SDN environments.

5. CONCLUSION

This study demonstrates the transformative potential of embedding neural network-based predictive intelligence within SDN for autonomous QoS optimization. By leveraging a LSTM model to capture temporal dependencies in network traffic, the system enables proactive, anticipatory control that significantly enhances routing efficiency and traffic management. Unlike static or classical ML approaches, this predictive framework allows SDN controllers to make informed decisions before congestion occurs, translating global network visibility into optimized flow engineering. The study establishes several key contributions:

(i) introducing an LSTM-enhanced SDN architecture capable of real-time predictive QoS control with minimal controller overhead, (ii) demonstrating superior network performance across heterogeneous traffic types, and (iii) providing a methodological framework for integrating DL into operational SDN environments. Operationally, the approach supports practical deployment in dynamic, high-demand networks, offering improvements in latency, throughput, and packet loss while remaining resource-efficient. Future research should address current limitations: enhancing interpretability to overcome the "black-box" nature of deep learning, improving resilience against adversarial traffic manipulations, and extending generalization across diverse topologies and IoT-intensive scenarios. Additionally, integrating XAI techniques will increase operator trust and facilitate wider industrial adoption. Ultimately, this work contributes to advancing SDN from programmable networks to intelligent, self-optimizing infrastructures, capable of delivering guaranteed QoS in the context of Industry 4.0, 5G, and emerging 6G networks, where real-time performance, adaptability, and reliability are paramount.

The proposed LSTM-based SDN framework demonstrates strong predictive QoS performance but faces several practical and theoretical limitations. Scalability is constrained under high flow rates; controller CPU and API bottlenecks limit responsiveness beyond ~12,500 concurrent flows, suggesting a need for distributed or edge-assisted inference architectures. Cold-start adaptation delays occur with unseen traffic types, requiring ~4 minutes to regain high accuracy, motivating exploration of online learning, few-shot, or meta-learning strategies. Measurement and control-plane latency further constrain real-time performance due to Open vSwitch counter noise and end-to-end rule installation delays, which may affect ultra-low-latency applications. Model interpretability remains limited as the LSTM operates as a "black-box," potentially hindering operator trust and adoption. Future research directions include integrating XAI techniques (e.g., shapley additive explanations (SHAP), local interpretable model-agnostic explanations (LIME)) for transparent decision-making, applying federated learning for privacy-preserving multi-domain deployment, and exploring hybrid LSTM–DRL architectures to combine short-term predictive control with long-term strategic resource optimization. Collectively, these directions aim to improve scalability, adaptability, transparency, and operational robustness of intelligent SDN control for diverse, high-demand network environments.

ACKNOWLEDGMENTS

This The authors would like to express their sincere gratitude to all colleagues and institutions that supported this research. This study did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

FUNDING INFORMATION

Authors state there is no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Muqamuddin Muhib	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Rangu Sridevi		✓				✓		✓	✓	✓	✓	✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

We have obtained informed consent from all individuals included in this study.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, upon reasonable request.

REFERENCES

[1] M. S. Abood *et al.*, "Improved 5G network slicing for enhanced QoS against attack in SDN environment using deep learning," *IET Commun.*, vol. 18, no. 13, pp. 759–777, Aug. 2024, doi: 10.1049/cmu2.12735

[2] M. J. Alenazi and J. Ali, "An effective deep-Q learning scheme for QoS improvement in physical layer of software-defined networks," *Physical Communication*, vol. 66, Art. no. 102387, 2024, doi: 10.1016/j.phycom.2024.102387.

[3] M. K. Awad, M. H. H. Ahmed, A. F. Almutairi, and I. Ahmad, "Machine learning-based multipath routing for software defined networks," *Journal of Network and Systems Management*, vol. 29, no. 2, Art. no. 18, 2021, doi: 10.1007/s10922-020-09583-4.

[4] E. H. Bouzidi, A. Outtagarts, and R. Langar, "Deep reinforcement learning application for network latency management in software defined networks," in *Proc. IEEE Global Communications Conf. (GLOBECOM)*, Waikoloa, HI, USA, Dec. 2019, pp. 1–6, doi: 10.1109/GLOBECOM38437.2019.9013221.

[5] E. H. Bouzidi, A. Outtagarts, R. Langar, and R. Boutaba, "Deep Q-network and traffic prediction-based routing optimization in software defined networks," *Journal of Network and Computer Applications*, vol. 192, Art. no. 103181, 2021, doi: 10.1016/j.jnca.2021.103181.

[6] A. Canovas, A. Rego, O. Romero, and J. Lloret, "A robust multimedia traffic SDN-based management system using patterns and models of QoE estimation with BRNN," *Journal of Network and Computer Applications*, vol. 150, Art. no. 102498, 2020, doi: 10.1016/j.jnca.2019.102498.

[7] D. M. Casas-Velasco, O. M. C. Rendon, and N. L. S. da Fonseca, "DRSIR: A deep reinforcement learning approach for routing in software-defined networking," *IEEE Trans. Network and Service Management*, vol. 19, no. 4, pp. 4807–4820, 2021, doi: 10.1109/TNSM.2021.3132491.

[8] J. Chen, C. Liao, Y. Wang, L. Jin, X. Lu, X. Xie, and R. Yao, "AQMDRL: Automatic quality of service architecture based on multistep deep reinforcement learning in software-defined networking," *Sensors*, vol. 23, no. 1, Art. no. 429, 2022, doi: 10.3390/s23010429.

[9] S. Faezi and A. Shirmarz, "A comprehensive survey on machine learning using in software defined networks (SDN)," *Human-Centric Intelligent Systems*, vol. 3, no. 3, pp. 312–343, 2023, doi: 10.1007/s44230-023-00025-3.

[10] H. Huang *et al.*, "A new fruit fly optimization algorithm enhanced support vector machine for diagnosis of breast cancer based on high-level features," *BMC Bioinf.*, vol. 20, no. 1, p. 290, Dec. 2019, doi: 10.1186/s12859-019-2771-z.

[11] M. A. Islam, R. Atat, and M. Ismail, "Software-defined networking based resilient proactive routing in smart grids using graph neural networks and deep Q-networks," *IEEE Access*, 2024, doi: 10.1109/ACCESS.2024.3438938.

[12] S. K. Keshari, V. Kansal, and S. Kumar, "A systematic review of quality of services (QoS) in software defined networking (SDN)," *Wireless Personal Communications*, vol. 116, no. 3, pp. 2593–2614, 2021, doi: 10.1007/s11277-020-07812-2.

[13] G. Kim, Y. Kim, and H. Lim, "Deep reinforcement learning-based routing on software-defined networks," *IEEE Access*, vol. 10, pp. 18121–18133, 2022, doi: 10.1109/ACCESS.2022.3151081.

[14] M. Latah and L. Toker, "Artificial intelligence enabled software-defined networking: A comprehensive overview," *IET Networks*, vol. 8, no. 2, pp. 79–99, 2019, doi: 10.1049/iet-net.2018.5082.

[15] G. J. Lin, C. F. Hung, and C. H. Ke, "A deep reinforcement learning-based bandwidth demand-oriented routing in software-defined networking," *ICT Express*, 2025, doi: 10.1016/j.icte.2025.07.009.

[16] K. Lu, Z. Du, J. Li, and G. Min, "Resource-efficient distributed deep neural networks empowered by intelligent software-defined networking," *IEEE Trans. Network and Service Management*, vol. 19, no. 4, pp. 4069–4081, 2022, doi: 10.1109/TNSM.2022.3218173.

[17] A. R. Mohammed, S. A. Mohammed, and S. Shirmohammadi, "Machine learning and deep learning based traffic classification and prediction in software defined networking," in *Proc. IEEE Int. Symp. Measurements & Networking (M&N)*, Catania, Italy,

[18] H. P. Nugroho, M. Irfan, and A. Faruq, "Software defined networks: A comparative study and quality of services evaluation," *Scientific Journal of Informatics*, vol. 6, no. 2, pp. 181–192, 2019, doi: 10.15294/sji.v6i2.20585.

[19] D. Nuñez-Agurto, W. Fuertes, L. Marrone, E. Benavides-Astudillo, C. Coronel-Guerrero, and F. Perez, "A novel traffic classification approach by employing deep learning on software-defined networking," *Future Internet*, vol. 16, no. 5, Art. no. 153, 2024, doi: 10.3390/fi16050153.

[20] M. F. Osman, M. R. M. Isa, M. A. Khairuddin, M. A. M. Shukran, and N. A. M. Razali, "A novel network optimization framework based on software-defined networking (SDN) and deep learning (DL) approach," *JOIV: International Journal on Informatics Visualization*, vol. 8, no. 4, pp. 2082–2089, 2024, doi: 10.62527/joiv.8.4.2169.

[21] L. L. Prasanth and E. Uma, "A computationally intelligent framework for traffic engineering and congestion management in software-defined network (SDN)," *EURASIP Journal on Wireless Communications and Networking*, vol. 2024, Art. no. 63, 2024, doi: 10.1186/s13638-024-02392-2.

[22] L. P. A. Sanchez, Y. Shen, and M. Guo, "DQS: A QoS-driven routing optimization approach in SDN using deep reinforcement learning," *Journal of Parallel and Distributed Computing*, vol. 188, Art. no. 104851, 2024, doi: 10.1016/j.jpdc.2024.104851.

[23] H. K. D. Sarma, "A survey on machine learning and deep learning based quality of service aware protocols for software defined networks," *Authorea Preprints*, 2021, doi: 10.36227/techrxiv.16950574.v1.

[24] N. Stylos and J. Zwiegelaar, *Big Data as a Game Changer: How Does It Shape Business Intelligence Within a Tourism and Hospitality Industry Context?* 2019.

[25] S. Shahzadi *et al.*, "Machine learning empowered security management and quality of service provision in SDN-NFV environment," *Computers, Materials and Continua*, vol. 66, no. 3, pp. 2723–2749, 2020, doi: 10.32604/cmc.2021.014594.

[26] G. Wassie, J. Ding, and Y. Wondie, "Detecting and predicting models for QoS optimization in SDN," *Journal of Computer Networks and Communications*, vol. 2024, Art. no. 3073388, 2024, doi: 10.1155/2024/3073388.

[27] D. Xia, J. Wan, P. Xu, and J. Tan, "Deep reinforcement learning-based QoS optimization for software-defined factory heterogeneous networks," *IEEE Trans. Network and Service Management*, vol. 19, no. 4, pp. 4058–4068, 2022, doi: 10.1109/TNSM.2022.3208342.

[28] J. Xu, J. Wang, Q. Qi, H. Sun, and B. He, "Deep neural networks for application awareness in SDN-based network," in *Proc. IEEE Int. Workshop on Machine Learning for Signal Processing (MLSP)*, Aalborg, Denmark, Sep. 2018, pp. 1–6, doi:

10.1109/MLSP.2018.8517088.
[29] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, O. Watanabe and T. Zeugmann, Eds. Berlin, Germany: Springer, 2009, pp. 169–178, doi: 10.1007/978-3-642-04944-6_14.
[30] Z. Yang, Q. Yang, and M. Yang, "Quality of service-oriented data optimization in networks using artificial intelligence techniques," *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 6, 2024, doi: 10.14569/IJACSA.2024.0150691.
[31] Y. Zhao, Y. Li, X. Zhang, G. Geng, W. Zhang, and Y. Sun, "A survey of networking applications applying the software defined networking concept based on machine learning," *IEEE Access*, vol. 7, pp. 95397–95417, 2019, doi: 10.1109/ACCESS.2019.2928564.
[32] G. Wassie, J. Ding, and Y. Wondie, "Traffic prediction in SDN for explainable QoS using deep learning approach," *Scientific Reports*, vol. 13, no. 1, Art. no. 20607, 2023, doi: 10.1038/s41598-023-46471-8.

BIOGRAPHIES OF AUTHORS

Muqamuddin Muhib is pursuing Ph.D. in the Department of Computer Science and Engineering at Jawaharlal Nehru Technological University Hyderabad, India. He completed his B.Tech. in Computer Science at Kabul Education University, Kabul, Afghanistan. He received his M.Tech. (CSE) From University College of Engineering, Osmania University, Hyderabad, India. His research areas include computer networks, software-defined networks, network security and neural networks. He worked as a lecturer in Faculty of Computer Science, Parwan University, Afghanistan. He can be contacted at email: Muqamuddin1989@gmail.com.

Rangu Sridevi works as Professor and Director, Directorate of Entrepreneurship, Innovation and Start-ups, JNTUH, Hyderabad, India. She received her B.E. (CSE) from Madras University and she received M.Tech. in CSE, from Andhra University. Subsequently she earned her Ph.D. in CSE, JNTUH, Hyderabad, India. Her research areas include network security and cryptography, cyber security, digital forensics and cyber crime investigation, computer networks, and data structures. She has published many research papers in conferences and reputed international journals. She can be contacted at email: sridevirangu@jntuh.ac.in.