IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 6 August 2023, accepted 15 August 2023, date of publication 28 August 2023, date of current version 18 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3309002

RESEARCH ARTICLE

Content-Aware Network Traffic Prediction Framework for Quality of Service-Aware Dynamic Network Resource Management

WAQAR ALI AZIZ[1,2], IACOVOS I. IOANNOU[1,2], (Member, IEEE), MARIOS LESTAS[3], HASSAAN KHALIQ QURESHI[4], ADNAN IQBAL[5], AND VASOS VASSILIOU[1,2], (Senior Member, IEEE)

[1]Department of Computer Science, University of Cyprus, 1678 Nicosia, Cyprus
[2]CYENS Center of Excellence, 2109 Nicosia, Cyprus
[3]Department of Electrical Engineering, Frederick University, 1036 Nicosia, Cyprus
[4]School of Electrical Engineering and Computer Science, National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan
[5]Department of Computer Science, Namal Institute, Mianwali 42250, Pakistan

Corresponding author: Waqar Ali Aziz (aziz.waqar-ali@ucy.ac.cy)

This work was supported in part by the European Union's Horizon 2020 Research and Innovation Program under Grant 739578; in part by the ADROIT6G Project of the Smart Networks and Services Joint Undertaking (SNS-JU) under Grant 101095363; and in part by the Government of the Republic of Cyprus through the Deputy Ministry of Research, Innovation and Digital Policy.

ABSTRACT Next-generation mobile networks, such as Fifth-Generation (5G), and Sixth-Generation (6G) are envisioned to undergo an unprecedented transformation from connected things to connected intelligence with more stringent characteristics, i.e., low end-to-end latency, high bandwidth, reliable connectivity, etc. Such networks will significantly increase network traffic in the distribution networks, causing the need for real-time automated decision-making, such as automated network resource allocation. The network resources be allocated dynamically based on the Quality of Service (QoS) requirements. However, the primary concern is that the distribution networks may get congested soon after allocating network resources. Thus, a QoS-aware prediction framework can be used proactively to predict the future trend of network traffic in each QoS class. In this paper, we propose a framework for predicting the heterogeneous multivariate QoS-aware network traffic to make the best use of network resources dynamically. Specifically, we use a Recurrent Neural Network (RNN) to integrate a Bidirectional Long Short-Term Memory (BLSTM) neural network. The results show that the fusion of RNN-BLSTM can predict the QoS-aware network traffic for over 13 hours with an average accuracy of 97.68%. Moreover, the proposed model is trained and tested over limited data, collected and identified through Deep Packet Inspection (DPI) over an operational network. In addition, we compared the RNN-BLSTM with other prediction algorithms (i.e., LSTM, ARIMA, SVM) in terms of precision, execution time, and energy consumption. Lastly, the proposed framework is used to assign the network resources to each QoS class based on the QoS requirements of that class and its pre-defined priority.

INDEX TERMS QoS, heterogeneous multivariate internet traffic, RNN-BLSTM, DPI.

I. INTRODUCTION

With the emergence of next-generation mobile networks, such as 5G and 6G, and with the ease of internet connectivity, users are growing exponentially and geographically [1]. The internet and its applications have already become the primary communication tool for daily tasks that affect our daily routine. To make use of the increased network resources promised by next-generation mobile networks (1 Tbps general peak data rate, 1 Gbps per user), new high-demanding applications, such as holographic teleportation and virtual and augmented reality (VR and AR) are emerging [2].

The associate editor coordinating the review of this manuscript and approving it for publication was Xujie Li.

These applications generate a vast amount of traffic with different characteristics and requirements regarding network resources, as shown in TABLE 1.

However, the increased number of services and applications poses a challenge for mobile networks' management and orchestration function [3]. Automated orchestration solutions in next-generation mobile networks should be set in place, implementing zero-touch operations [4]. Managing resources in the next-generation mobile networks must rely on machine learning techniques and provide optimized solutions [5]. The European Telecommunications Standards Institute (ETSI) group Zero-touch network Service Management (ZSM) and ETSI group Experimental Networked Intelligence (ENI), each has designed a reference architecture for the management of next-generation mobile networks using artificial intelligence techniques and context-aware policies [6], [7]. Although network automation is a key point in next-generation mobile networks, the aforementioned architectures do not address network slicing in detail, which directly affects network scalability [8]. Thus, a framework is needed to manage the core network resources dynamically in order to handle the diverse internet applications that heavily depend upon the next-generation mobile network resources (as highlighted in [9]).

Performing network traffic prediction and resource allocation without incorporating the application-level information can affect the end-user's Quality of Experience (QoE) [10] because different applications have different QoS requirements [11]. Network resource allocation can be fixed or dynamic [12]. The former method leads to under-utilization or over-utilization of network resources. Based on the QoS requirements, the network resources can be allocated dynamically to the internet applications [13]. However, the prediction and the allocation of network resources to meet the QoS demands of the applications individually is not a feasible solution. Therefore, applications with similar QoS requirements should be clustered into one QoS class [14]. Predicting traffic and allocating resources to each QoS class proactively is realizable. It can reduce the impact of network bottlenecks for delay-sensitive applications and improve the end user's QoE.

The network traffic can be identified through the payload-based technique [15]. The identified traffic can be characterized into the QoS classes based on the application layer information [16]. The network traffic can be predicted with the help of a deep learning-based prediction model within each QoS class, which may traverse through the considered network for the considered time in the future [17]. Then, based on the predicted traffic and the priority of the QoS class, the network resources can be allocated proactively and dynamically to minimize the network's congestion impact for delay-sensitive applications.

In [18], the authors propose a framework to allocate the network resources to the video streams proactively and dynamically. In [19], network resources are reallocated based on the traffic prediction by categorizing the network traffic into k QoS classes. Still, there is significantly less discussion on the prediction model and the criteria to create k QoS classes of the network traffic. In [20], the authors use an Auto-Regressive Integrated Moving Average (ARIMA) model (discussed in Sect. II-A) to predict the traffic for QoS flow routing. However, the accuracy of machine learning and deep learning regression models is far better than the ARIMA model, especially for multivariate time series prediction [21]. This motivates us to study the recurrent neural networks to propose a content-aware multivariate network traffic prediction framework for the dynamic allocation of network resources to the traffic inside each QoS class. Our work shares similarities with [22], where both studies address the problem of content-aware network traffic prediction. However, our approach differs in the use of a prediction model. We also propose using a dynamic resource allocation algorithm based on the predicted network traffic inside each QoS class.

TABLE 1. QoS demands of different internet applications [11].

Applications	Bandwidth	Sensitive to		
		Delay	Jitter	Loss
Voice over Internet Protocol	Low	High	High	Med
Video Conferencing	High	High	High	Med
Streaming Video	High	Med	Med	Med
Streaming Audio	Low	Med	Med	Med
Client/Server Transactions	Med	Med	Low	High
Email	Low	Low	Low	High
File Tranasfer	Med	Low	Low	Hig

Accurate prediction of multivariate network traffic is still a significant research challenge, especially QoS-aware multivariate heterogeneous traffic prediction. This is because traffic inside each QoS class has distinct traffic patterns. These traffic patterns make the network traffic data highly heterogeneous. Learning and predicting this heterogeneous data in an online network is challenging and requires vast training data. Furthermore, the datasets provided in the literature do not provide application-level information on internet traffic, e.g. [23]. This paper aims to provide a framework that can help network service providers to allocate network resources dynamically and proactively based on the amount of traffic inside each QoS class, to minimize the impact of network congestion for delay-sensitive applications.

The proposed framework aims to solve the problem of proactive QoS-aware network resource management and involves integrating multiple components, such as network traffic monitoring, network traffic categorization, network traffic prediction in each QoS class, and network resources allocation to the traffic inside each QoS class. In this study, we continue our previous efforts specific to identifying and predicting multimedia traffic [24]. More specifically, the foundation of the paper's investigated approach lies in; the accurate identification of network traffic using deep packet inspection, classification of network traffic, prediction of network traffic through machine learning-based high-performance approaches (e.g., RNN-BLSTM), and the Proactive QoS-aware network resource allocation based on

the outcomes of prediction. Thus, the key contributions of this paper are:

1) Content identification using DPI by analyzing the packet payload.
2) Stationarity testing of the captured time series for the data validation.
3) Categorization of captured network traffic into one of six QoS classes based on the QoS requirements.
4) Prediction of network traffic in each QoS class using the proposed multivariate time series forecasting model, RNN-BLSTM.
5) Evaluation of the proposed RNN-BLSTM model using a limited dataset obtained in the first step.
6) Comparison of the proposed prediction model with others such as ARIMA, SVM, and LSTM in terms of prediction accuracy, latency, and power consumption.
7) Validation of the proposed prediction model on the Telecom Italia dataset.
8) Dynamic network resources allocation to each QoS class based on the predicted amount of network traffic and the priority of QoS class.

According to our analysis, the RNN-BLSTM has never been employed for heterogeneous multivariate content-aware network traffic prediction except in speech gender classification [25], rapid speaker adaptation [26], multi-pitch estimation [27] and daily activity recognition [28].

The rest of the paper is organized as follows. Section II provides the background information and related work on approaches associated with network traffic prediction. The problem description is shown in Section III. Section IV provides a framework for data capture, data validation, traffic prediction using deep neural networks, and network resource allocation. The experimental setup and the performance evaluation metrics are presented in section V. Section VI offers a comprehensive performance evaluation of the proposed framework. Finally, we conclude the paper in Section VII.

II. BACKGROUND INFORMATION AND RELATED WORK

This section provides background information on the investigated approaches used for traffic prediction and related work on traffic prediction.

A. BACKGROUND INFORMATION

This section provides a brief introduction to the approaches used in the paper. More specifically, the background information about ARIMA, SVM, RNN, LSTM, and the BLSTM are comparable with the proposed RNN-BLSTM approach.

1) AUTO-REGRESSIVE INTEGRATED MOVING AVERAGE (ARIMA)

ARIMA [29], [30], [31], [32] is a generalization of a model that combines Auto-Regressive (AR), and Moving Average (MA) processes to generate a composite model of time series. ARIMA (p, d, q) captures the model's fundamental parts. ARIMA is the shorthand for the general form of the AR-I-MA model (p, d, q), as the acronym suggests:

- AR regression model leverages the inter-dependencies between a single observation and several lagged observations (p).
- Integral (I) makes a time series stationary (d) is calculating the differences between observations at various intervals.
- MA is a technique that accounts for the inter-dependence between observations and residual error components (q). A moving average model is applied to lag observations.

Additionally, non-seasonal short-term components likely contribute to the model when seasonal time series data are used. Consequently, estimating the seasonal ARIMA model, which incorporates both non-seasonal and seasonal components, is essential. That is why ARIMA can be represented with the following form and ARIMA(p, d, q) (P, D, Q)[1] is using the data's time plot. For example, it should utilize variance-stabilizing transformations and differences if the variance grows with time. Moreover, ARIMA uses the Autocorrelation Function (ACF) to measure the linear dependence between observations in a time series that are separated by a lag p, the Partial Autocorrelation Function (PACF) to determine how many autoregressive terms q are required, and the Inverse Autocorrelation Function (IACF) for detecting over difference, the preliminary values of the autoregressive order p, the order of difference d, and the moving average order q. The d denotes the frequency order of the transition from non-stationary to stationary time series [33], [34].

2) SUPPORT VECTOR MACHINE (SVM)

SVM is a supervised learning model with an associated learning algorithm that performs classification and regression analysis in machine learning. SVM is among the most accurate prediction methods based on the statistical learning framework or Vapnik Chervonenkis (VC) theory. SVM assigns training examples to points in space to maximise the separation between classes. Then, the category membership of new examples is predicted based on which side of the gap they fall on. Given a set of training examples, each of which is labelled as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one of the two categories, thereby creating a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). SVM can efficiently perform non-linear classification in addition to linear classification by implicitly mapping their inputs into high-dimensional feature spaces using the kernel function [35].

[1]The S is the generic form of a seasonal ARIMA model, where p is the non-seasonal AR order, d is the non-seasonal difference, q is the non-seasonal MA order, P is the seasonal AR order, D is the seasonal difference, Q is the seasonal MA order, and S is the duration of the recurring seasonal pattern. Determining the values of (p, d, and q) is the most critical step in estimating the seasonal ARIMA model (P, D, Q).

3) RECURRENT NEURAL NETWORK (RNN)

RNN [36] is a deep neural network in which the connections between the nodes can create a cycle, permitting the output of some nodes to affect subsequent input to the same nodes. This enables its temporally dynamic behaviour. RNN, built from feed-forward neural networks, can manage input sequences of different lengths by utilizing their internal state (memory). A recurrent neural network describes networks with an infinite impulse response, while a ''convolutional neural network'' describes networks with a limited impulse response. Both network types exhibit dynamic temporal behaviour. If RNN has time delays or feedback loops, the store may be replaced by a different network or graph.

RNN range from partially linked to fully connected, in addition to two RNNs with a single layer. Comparable to a three-layer neural network, the Elman network stores the outputs of the hidden layer in context cells. The output of a context cell is returned to the hidden neuron together with the signal's origin. Each neuron in the hidden layer receives input from both the input layer and context cells. Elman networks[2] can be trained with standard error back-propagation, with the context cell output regarded as an additional input. RNNs are dynamic systems with an internal state at each categorization time step. This is due to the circular connections and potential self-feedback connections between neurons in the upper and lower layers. These feedback links allow RNNs to convey data from past occurrences to processing steps that are now occurring. Therefore, RNN constructs a memory of time series events [37].

4) LONG SHORT-TERM MEMORY NEURAL NETWORK (LSTM)

LSTM [36], [38], [39], [40], [41] is a deep neural network with feedback connections. LSTM can learn how to bridge short temporal lags of over a thousand distinct time steps. The LSTM architecture is designed to provide RNN with a short-term memory that can endure for thousands of timesteps, hence the name ''long short-term memory''. A standard LSTM unit comprises a cell, an input gate, an output gate, and a forget gate. The network's connection weights and biases change once per training session, analogous to how physiological changes in synaptic strengths store long-term memories; the network's activation patterns change once per time-step, analogous to how the moment-to-moment change in the brain's electric firing patterns stores short-term memories. Long-term, short-term memory is a gradient-based method that circumvents the vanishing error problem caused by the derivative of the activation function used to create the neural network (LSTM). The system utilises Constant Error Carousels (CECs), guaranteeing a steady flow of mistakes within specific cells.

Multiple gate units control cell access by knowing when to allow access. This is the crucial property of LSTM, which allows for the long-term storage of short-term memory. This is when the various LSTM network components come into play, as we must still handle the connections from other units to unit u. LSTM resolves the problem of conflicting weight updates by extending the CEC with input and output gates connected to the network's input layer and other memory cells. This produces a memory block, which is a more complex LSTM unit. The input gates, which are simple sigmoid threshold units with an activation function range of [0, 1], scale the signals from the network to the memory cell; activation is close to zero when the gate is closed. In addition, they can learn how to protect the information stored in u from interference by outside signals. The activation of a CEC by the input gate defines the cell state. The cell remembers values across arbitrary time intervals, and the three gates control data flow into and out of the cell. The output gates can learn how to regulate access to the memory cell's contents, thereby buffering other memory cells from u-caused disturbances. Consequently, the primary function of multiplicative gate units is to grant or deny access to continuous error flow through the CEC.

LSTM neural networks can be used in a wide range of applications. In [42], the authors suggest a framework called COME-UP computation Offloading for mobile edge computing, using LSTM-based user direction prediction. Utilizing a feed-forward technique, the LSTM trains the learning model and predicts the subsequent position by taking into consideration the location, velocity, and direction of the previous mobility. In [43], the authors provide a QoS prediction framework that is based on an LSTM architecture and is capable of forecasting QoS metrics for connected and automated vehicles.

5) BI-DIRECITONAL LONG SHORT-TERM MEMORY NEURAL NETWORK (BLSTM)

In [44], authors analyze the possibility of analysing a given location's future and history using LSTM. The input is sent in both directions to two independent LSTM networks coupled to the same output layer. When categorising phonemes, bidirectional training provides an architectural advantage over unidirectional training. Bidirectional LSTM avoids the initial LSTM one-step truncation and calculates the complete error gradient. This method simplified the development of bidirectional LSTM and made it possible to train it with standard Back Propagation Through Time (BPTT), because typical RNN processes each point in a sequence by analysing only one direction: the past. Additionally, the BLSTM neural network has the ability to keep track of information, making it perfect for processing time series sequential data [45].

6) RNN-BLSTM

RNN-BLSTM [46], [47] combines two learning algorithms: Real-Time Recurrent Learning (RTRL) training network components (in this component, the training network is deriving a gradient-based update rule for recurrent networks)

[2]The Elman network is typically a two-layer network with feedback from the first layer's output to the first layer's input.

before cells and BPTT teaching network components (in this component a gradient-based technique for training certain types of recurrent neural networks is utilised) after cells. This combination is used to preserve CEC in BLSTM memory block cells that employ the original formulation of BLSTM. The latter units are compatible with RTRL since some partial derivatives (related to the state of the cell) must be computed at each step whether a goal value is provided. For now, we only allow the gradient of the cell to propagate across time, therefore severing the gradients of the other recurrent connections.

RNN-BLSTM excels at tasks that need the long-term storage of a modest quantity of data. This attribute results from the utilization of memory blocks. Memory blocks are fascinating: they have input and output gates prohibiting superfluous data from entering or leaving the memory block. Memory blocks are also fitted with a forget gate that weighs the information inside the cells so that when prior information becomes irrelevant for some cells, the forget gate can reset the state of the individual cells within the block. In addition to providing continuous prediction, forget gates can cause cells to forget their previous state, eliminating bias in prediction.

The RNN-BLSTM requires a specified network configuration. Since the number of memory blocks does not fluctuate dynamically, network memory is ultimately constrained. This constraint is unlikely to be addressed by uniformly increasing the network size, indicating that modularisation improves successful learning. However, the process of modularisation is ''not generally visible''.

B. RELATED WORK

To increase the accuracy of network traffic forecasts, the authors in [48] propose an ensemble method for time series prediction based on the LSTM neural network. The Adaptive Boosting (AdaBoost) technique is then used to improve the LSTM method because it is the most common boosting algorithm. The authors ultimately revised AdaBoost and combined it with LSTM. The resulting AdaBoost-LSTM method is more precise than the LSTM algorithm alone, and in some situations even more accurate than ARIMA. The AdaBoost-LSTM algorithm is used to estimate Internet traffic.

Several early studies characterise network traffic forecasting as a straightforward univariate time series forecasting problem, neglecting the network traffic matrix structure. Deep learning algorithms have proven more effective for anticipating network traffic than linear and statistical methods. The authors in [49] model the problem of network traffic matrix prediction as a video prediction assignment. A Convolutional LSTM-based (ConvLSTM) sequence-to-sequence model, named as ConvLSTM-TM, is proposed for traffic matrix predicting in the subsequent time slot. ConvLSTM-TM outperforms five deep-learning baselines on three real-world traffic matrix datasets regarding prediction error.

The authors in [50] utilize the LSTM neural network model to forecast non-linearly behaving network traffic. To improve the accuracy of the prediction model, an auto-correlation coefficient is introduced based on auto-correlation features. Conventional network models cannot predict network traffic that exhibits non-linear system behavior, and results show that LSTM works well to predict network traffic. Further, GA-LSTM, an LSTM method based on a Genetic Algorithm (GA), is proposed in [51] for predicting network traffic; experimental results indicate that the proposed GA-LSTM achieve higher prediction accuracy with minor prediction errors.

Authors in [33] investigate the use of the ARIMA model, whereas in [52] show the effectiveness of an SVM model for network traffic prediction. Most of the research work listed in [23] is based on univariate time series prediction and is carried out on the dataset listed in [53]. However, the dataset does not provide any information about the application-level contribution of network traffic. The accuracy of prediction models is better on univariate time series [54], [55], [56] compared to multivariate time series dataset [57] which can be improved. A comparison of deep learning prediction approaches on the network traffic is presented in [58].

In [19], the authors study the impact of traffic estimation on network resource allocation. They propose a QoS-aware resource allocation mechanism using traffic prediction in software-defined cloud networks with the help of k different traffic classes with various QoS requirements, but they did not explain the model used to predict the traffic inside the QoS classes. In [59], a multi-mode green IoT in smart park multi-timescale resource scheduling and route management optimization challenge was investigated using Software Defined Networking (SDN) and Network Function Virtualization (NFV).

Tomovic et al. in [60] considers the problem of minimizing Maximum Link Utilization (MLU) by periodic reconfiguration of load balancing weights for the links. Traffic forecasting is implemented to reduce the frequent reconfiguration over a predicted time horizon, but the paper does not provide any detail about the model used for the prediction of traffic in each QoS class. Along with that, there is no study on the energy consumption, execution time, latency time of the prediction, and the accuracy of the model used for the QoS-aware multivariate time series prediction.

Most of the aforementioned work mainly focuses on the univariate network or cellular traffic prediction (e.g. total internet traffic) without considering the QoS classes. Our proposed framework in this paper differs from the above works, since we solve the QoS-aware traffic prediction problem to allocate the network resources proactively to avoid congestion in the network for high-priority traffic.

III. PROBLEM DESCRIPTION

This paper considers a network where the traffic is identified and classified into p QoS classes. Let the traffic observation

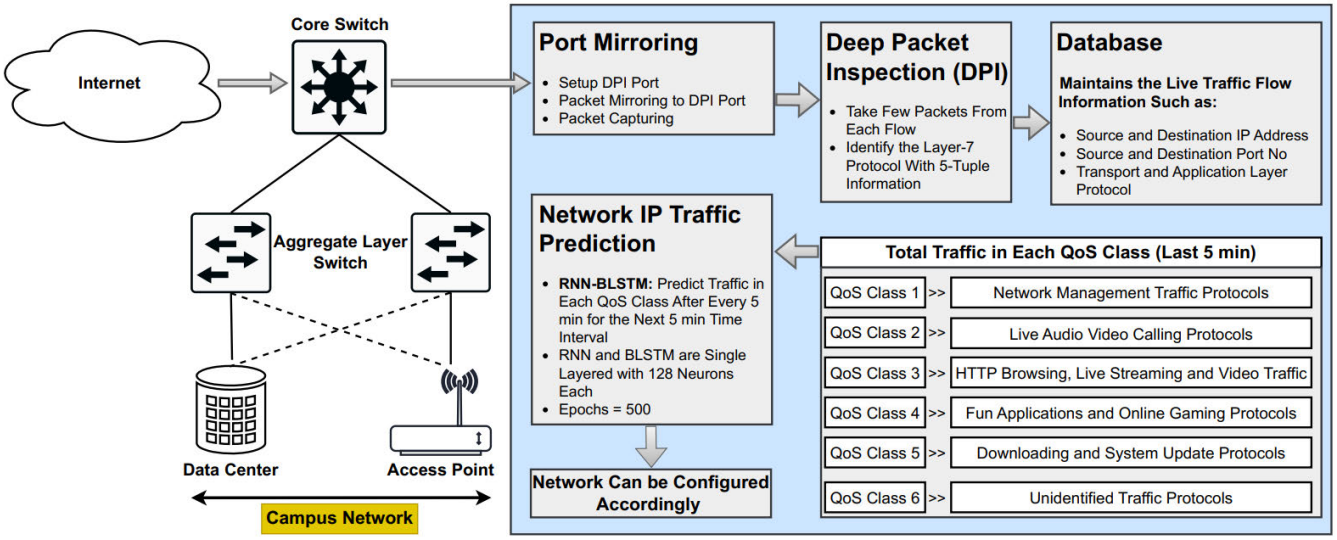

FIGURE 1. System model for the multivariate heterogeneous network traffic prediction framework.

inside each QoS class be over the past K time slots. Input vector $\boldsymbol{X}_t = \{\boldsymbol{x}_{(t)1}, \boldsymbol{x}_{(t)2}, \ldots, \boldsymbol{x}_{(t)p}\}$ shows the amount of traffic volume for p QoS class at time t. Here our objective is to find the traffic volume in each QoS class for the next time stamp $\hat{\boldsymbol{Y}}_{(t+1)} = \{\hat{\boldsymbol{y}}_{(t+1)1}, \hat{\boldsymbol{y}}_{(t+1)2}, \ldots, \hat{\boldsymbol{y}}_{(t+1)p}\}$ such that:

$$\min \lim_{T\to\infty} \frac{1}{T} \sum_{t\in 1}^{T} \boldsymbol{L}(\hat{\boldsymbol{Y}}_{(t+1)}, \boldsymbol{Y}_{(t+1)})$$

where L represents the loss function that measures the difference between the predicted $\hat{\boldsymbol{Y}}_{(t+1)}$ and the actual network traffic $\boldsymbol{Y}_{(t+1)}$. T represents the total time steps, $T = \{1, 2, \cdots, t, T\}$, where two-time steps have the same time interval (for example, 5 minutes), which is used to monitor the network's state and generate predictions about the future. During network congestion, fair bandwidth allocation can affect the network traffic, which is sensitive to delay and packet loss. Hence, the next step is to dynamically assign network resources to each QoS class based on the predicted traffic and the priority of that class.

IV. QoS AWARE PREDICTION FRAMEWORK

This section describes the proposed framework in the following manner: i) Capturing of network traffic using port mirroring technique; ii) Setting up the database to store the captured network traffic and its validation through unit root test; iii) Categorizing the captured network traffic into one of six QoS classes; iv) Predicting the traffic in each QoS class using the proposed RNN-BLSTM neural network model; and v) allocating the network resources based on the traffic priority and the predicted traffic volume, constrained by the available data rate, using the designed algorithm.

Our proposed framework shown in Fig. 1 contains the following modules:

1) DPI engine to accurately identify the network traffic.
2) Database to store the statistics from the DPI engine to classify the incoming network traffic into the respective QoS classes.
3) Prediction module to predict the future volume in each QoS class.
4) Algorithm to dynamically allocate the available network bandwidth based on the predicted traffic and the priority of each QoS class.

In the description of each module, we are using a dataset collected from the university network, shown in Fig. 2.

A. IDENTIFICATION OF NETWORK TRAFFIC VIA PORT MIRRORING

Accuracy and throughput are the two major challenges in the identification of live network traffic [61], [62]. There are mainly three techniques for network traffic identification which are [63]:

1) Port-based approach
2) Statistical analysis-based approach
3) Payload-based approach

A port-based approach can easily identify the network protocols such as Hyper Text Transfer Protocol (HTTP), File Transfer Protocol (FTP), etc. The port-based technique is efficient in terms of latency, but it is frequently unreliable due to the emergence of dynamic port numbers. In a statistical analysis-based approach, a machine learning algorithm is used to classify the incoming network traffic into different QoS classes [64]. This technique is not as accurate as a payload-based approach but provides better throughput. However, this technique cannot identify specific application types. Along with that, it is computationally intensive [65]. In the payload-based approach (i.e. DPI), a few initial packets per flow are analysed. The number of packets depends on

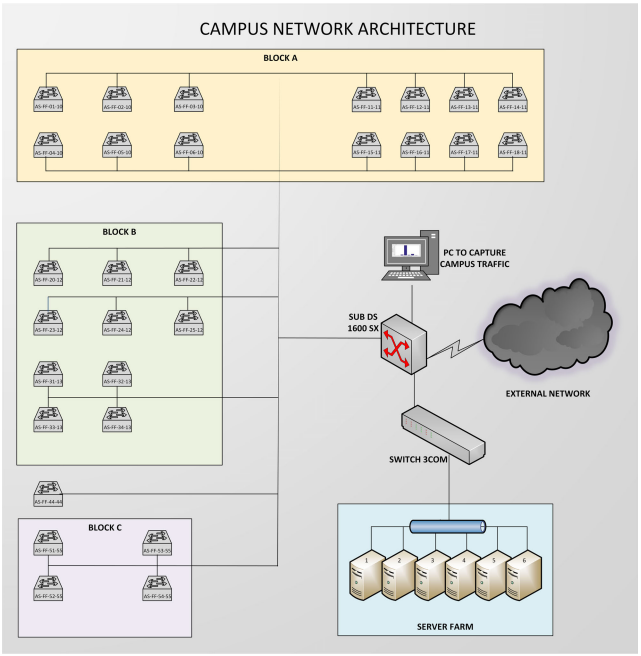

FIGURE 2. Data collection setup.

the type of application and on the efficiency of the DPI algorithm [66]. For example, network management protocols can easily be identified using a single packet [67].

Besides, the DPI technique can also be used for the network optimization frameworks where application layer information is required [67]. The DPI algorithm [68] used in our proposed methodology requires one to eight packets per flow, depending on the application type. It can also monitor up to 8.5 Gbps traffic on commodity hardware and identify more than 332 well-known application layer protocols (that use maximum network bandwidth).

Our framework aims to reduce network latency and increase the precision of network traffic identification. Thus, we employ port mirroring [69] to redirect all the network traffic to a specific switch port (a DPI port). With this method, we duplicate the captured packets and transfer them to the DPI engine without packet loss as shown in Fig. 2.

B. DPI ENGINE

This section examines the performance of the DPI engine, which is a major component in the current and future evaluation, and for the identification and classification of network traffic. The efficiency of the DPI engine is a crucial factor while identifying internet traffic. Thus, Fig. 3 shows the throughput of the DPI engine in terms of packet drop rate vs the packet inter-arrival time. The capability of the DPI engine is evaluated by passing self-generated User Datagram Protocol (UDP) packets of variable sizes, including 60, 1000, and 1500 Bytes. The packet drop rate varies with the size of the packet. The graph indicates a zero drop rate at the beginning. Still, the onset of a phase transition of the packet with the size of 1500 Bytes can be identified at 110μsec packet inter-arrival time. In contrast, the packet with the

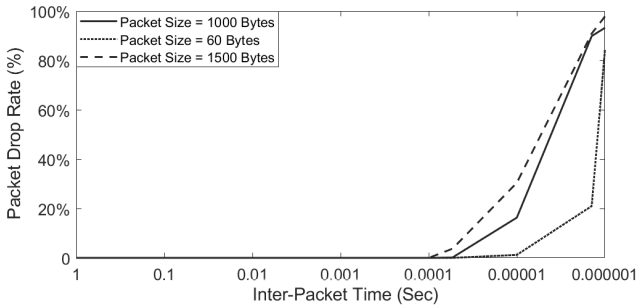

FIGURE 3. DPI engine efficiency in terms of packet drop rate at different inter-arrival times of the packets.

TABLE 2. Unit root test statistics.

QoS Class	Test Critical Values			Level		First Difference	
	1% Level	5% Level	10% Level	t-Stats	P-Value	t-Stats	P-Value
1	-3.9838	-3.4224	-3.1341	-0.8608	0.9578	-17.5973	0
2	-3.9839	-3.4225	-3.1341	-0.9313	0.9501	-7.8934	0
3	-3.9839	-3.4224	-3.134	-1.2235	0.9035	-7.6061	0
4	-3.9838	-3.4224	-3.1341	-0.4776	0.9842	-13.0475	0
5	-3.984	-3.4225	-3.1341	-1.5988	0.7919	-4.7134	0.0008
6	-3.984	-3.4225	-3.134	-0.5423	0.9812	-10.7258	0
Total	-3.9839	-3.4225	-3.1341	-1.5689	0.8034	-4.9931	0.0002

size of 60 and 1000 Bytes show packet drop at 82 μsec and 86 μsec packet inter-arrival time, respectively. The DPI engine works as a separate module, as shown in the system model in Fig. 1, which does not affect the throughput of the network devices.

C. DATABASE AND DATA VALIDATION

In this sub-section, we provide the details about the database created by the statistical information generated by the DPI engine (shown in section IV-B) and the procedure to validate the captured time series dataset.

1) DATABASE

A database is required to store and process the statistical information generated by the DPI engine. For this purpose, a relational database is connected to the DPI engine via a particular API that supports tabular data format and is updated after every 10 seconds. Entries are made based on per-flow information. The database maintains application layer protocol, 5-tuple information (source IP address, source port, destination IP address, destination port, transport protocol), total volume, and packets per flow. This information is enough to classify the network traffic into different QoS classes.

2) DATA VALIDATION

Our proposed framework requires application layer information to classify the network traffic into different QoS classes. The dataset listed in the literature, e.g. [23] does not provide information about the application layer information in network traffic. In this sub-section, we validate the dataset captured by using the network shown in Fig. 2. The time series validation is done by checking the stationarity using the Augmented Dickey-Fuller (ADF) method, where the Unit root test is used to check the existence of unit roots in the

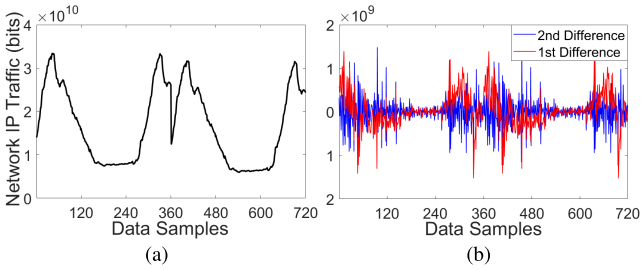

FIGURE 4. (a) The network traffic pattern with non-constant mean, (b) first and second level difference of network traffic pattern.

Algorithm 1 Classification of Network Traffic:

Result: Classification of network traffic into six QoS classes based on the application layer protocol.

t : Initialization time

QoS class i where i = Constant, valued between 1 - 6

Initialization ;

$t = 0$;

while t = *5 minutes* **do**

  Go to the database;

  if *if application protocol* $\in$ *QoS class i* **then**

    Put the flow in QoS class i

    Calculate the total traffic in QoS class i

  else

    Update Unidentified QoS class 6

  end

end

time series. As shown in Table 2, the captured time series is not stationary at the level but stationary at the first difference, assuming the threshold P-value is less than 0.05. The value of t-stats less than the test critical value rejects the hypothesis that there is a unit root in the time series, showing that the time series is stationary. The non-stationarity of the time series can be visualized from Fig. 4 as the mean of the time series is not constant. The unit root test result with the first and second difference is depicted in Fig. 4(b). It is observed that the time series become more stationary at the second difference with a constant mean. However, Table 2 shows that the series is still stationary at the first difference, which enables us to provide a fair comparison of prediction algorithms (ARIMA) with the proposed prediction model.

D. CLASSIFICATION OF INTERNET TRAFFIC

Classification of the network traffic provides a comprehensive image of the network. In our proposed framework, Algorithm 1 is used to classify the network traffic into six QoS classes based on the application layer protocol. We assume that similar applications have similar network resource requirements.

Network management protocols may experience latency due to other traffic flows, particularly during network congestion, which may cause network configuration delays. In prior research, network management protocols are not

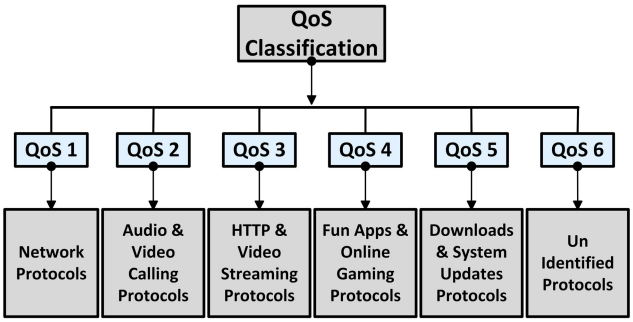

FIGURE 5. Classification of network traffic into different QoS classes.

treated as a distinct QoS class [19], [60], [70]. As illustrated in Fig. 5, we use six QoS classes, and the network management protocols are assigned the highest priority. Based on the QoS demands on the network side and QoE demands on the user side, the priorities of QoS classes can be modified. Furthermore, the algorithm also calculates the aggregated network traffic volume inside each QoS class on 5 minutes time intervals.

Fig. 6(a) shows the total data volume in bytes, whereas Fig. 6(b) shows the total number of packets in each QoS class captured for 48 hours. It reveals that network management traffic has a total volume of 18.04 GB inside total traffic. Live audio/video calling applications generated 92.1 GB of traffic during 48 hours. These traffic samples will be used as a training and test dataset for the proposed RNN-BLSTM prediction model. Browsing traffic, including live streaming and video traffic, has 209.1 GB, Entertainment and gaming traffic has 172.8 GB, and Downloading and system update traffic has 220.1 GB. In our captured traffic, 14.4 GB of traffic remains unidentified in the total network traffic. This categorization of network traffic into QoS classes is based on application layer protocol.

The number of iterations in the while loop determines the algorithm's total complexity. Initialization of the variables takes constant time $\mathcal{O}(1)$.[3] Accessing the database for n traffic flows depends on the size of the database, denoted by $\mathcal{O}(n)$. Checking the application layer protocol and the QoS class for each traffic flow takes a constant amount of time $\mathcal{O}(1)$. Assigning a QoS class to each traffic flow and calculating the total network traffic in that QoS class take $\mathcal{O}(1)$ time. Finally, updating the unidentified QoS class takes $\mathcal{O}(1)$ time. Hence the overall time complexity of the given Algorithm 1 can be determined as:

$$\text{Time Complexity} \approx \mathcal{O}(n) + 4\mathcal{O}(1) = \mathcal{O}(n)$$

E. PREDICTION MODEL

RNN-BLSTM combines two learning algorithms, (1) Real-Time Recurrent Learning (RTRL), which is based on the

[3] Big O Notation is a metric for determining the efficiency of an algorithm. It provides an estimate of how long it takes your code to execute on various inputs. You can also use it to determine how well your code scales as the magnitude of the input increases [71].

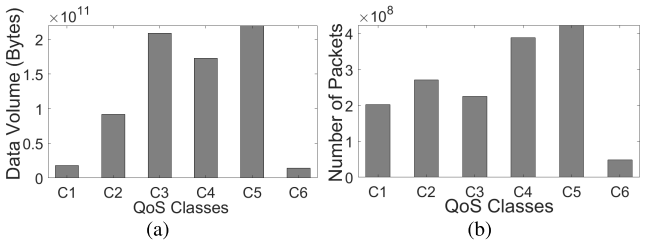

FIGURE 6. (a) Total traffic volume in each QoS class, (b) total network packets in each QoS class.

gradient-based update rule, and (2) Bidirectional LSTM that consists of two separate hidden layers, the forward (resp. backwards) layer, which learns on past (resp. future) trend. These two hidden layers are connected with the output layer to fuse past and future trends. This combination achieves better prediction accuracy on long-term dependencies. Fig. 7 shows the proposed neural network architecture.

1) INPUT LAYER

The dataset comprises $n = 720$ data samples, with $p = 6$ QoS classes equivalent to 6 input features. A batch size of $n_b = 32$ is used for the training of the proposed model by keeping a balance between the estimation of accurate gradient and the computation cost. Moreover, the timestamp $K = 5$ is used.

2) RECURRENT NEURAL NETWORK

In our proposed mechanism, a single-layered RNN is used which is comprised of $m_1 = 128$ hidden units. The input size of the RNN layer is $n_b \times K \times p = 32 \times 5 \times 6$. This RNN layer produces the output equal to $n_b \times K \times m_1 = 32 \times 5 \times 128$. The single-layered RNN can be formulated as follows:

$$h_{(t-k)} = f(X_{(t-k)}, h_{(t-k-1)}), k \in \{0, 1, 2, \ldots, K-1\} \quad (1)$$

where $f(.)$ is the recursive transfer function applied on the input data window. At the time t, the RNN maps an input sequence $\boldsymbol{X_t}$ to the corresponding sequence of output O_t. By considering the hidden state transition from $h_{(t-k-1)} \rightarrow h_{(t-k)}$ and by using (1), the recurrent state of the RNN is expressed as:

$$h_t = \tanh\left[W_h^{(1)} h_{(t-1)} + W_x^{(1)} X_t + b_h^{(1)}\right] \quad (2)$$

where, tanh is the activation function equal to $\frac{exp(x)-exp(-x)}{exp(x)+exp(-x))}$, W_h and W_x are the weight vectors at the recurrent neuron and input neuron, respectively. The previous state is denoted as $h_{(t-1)}$ whereas b_h is the bias vector at the current recurrent state. The superscript shows the i^{th} layer[4] in the RNN model. The output of the recurrent state at time t is denoted as:

$$Z_t = \text{SoftMax}[W_o^{(1)} h_{(t)} + b_o^1] \quad (3)$$

In the above equation, W_o is the weight vector at the output neuron, h_t is the recurrent state, and b_o is the bias at the output neuron. Finally, SoftMax = $\frac{e^{x_i}}{\sum_{j=1} e^{x_j}}$ is used as a final activation function to normalize each output of the recurrent state to a probability distribution over the predicted target values. Z_t becomes an input to the single-layered BLSTM with the return sequence = True. The working principle and analysis of the softmax activation function in the back-propagation neural networks is explained in [72].

[4]In the proposed model, single-layered RNN is applied (i=1).

3) BIDIRECTIONAL LONG SHORT-TERM MEMORY NEURAL NETWORK

BLSTM is designed in such a way that it consists of 3 gates to control the flow of information, which are:

1) Forget gate
2) Input gate
3) Output gate

In the BLSTM technique, the input sequence is first fed to LSTM forward cell $\overrightarrow{O} = \text{LSTM}_f(Z_t)$ and then to backward cell $\overleftarrow{O} = \text{LSTM}_b(Z_t)$ (which can be seen in the weight vector of BLSTM layer, 2∗128). The output is the combination of the output of two cells $O = [\overrightarrow{O}, \overleftarrow{O}]$. This forward and backward flow of the sequential data results in better and faster learning. BLSTM layer takes input of size $n_b \times K \times m_1 = 32 \times 5 \times 128$ and produces output of size $n_b \times K \times (2 * m_2) = 32 \times 5 \times (2 * 128)$.

At time t, the output of the RNN layer is $Z_t = \{z_{(t)1}, z_{(t)2}, \ldots, z_{(t)m_2}\}$ which is feeded into the BLSTM layer, while h_t represents hidden layer output. The cell input state is $\tilde{C}_t$, where C_t is the output state. f_t, i_t, and o_t show forget, input and output gate states, respectively, having the same dimensions. Both h_t, C_t are transferred to the next state. In (4), the σ represents the sigmoid activation function which is $\frac{1}{1+exp(-z)}$ and tanh represents the hyperbolic tangent function equals to $\frac{exp(z)-exp(-z)}{exp(z)+exp(-z))}$. The sigmoid function limits the prediction values between 0 and 1, whereas the tanh function regulates the network by limiting the values between -1 and 1. W_f, W_i, W_o, W_c are the weight matrices and b_f, b_i, b_o, b_c are the biases at each gate and cell input.

Forget gate decides which information should be added to the cell state. It has a value between 0 and 1. Where 0 represents the uselessness of the data from the previous state. Forget gate takes the output of the previous state h_{t-1} along with the current input z_t and then passes these values through the sigmoid function as shown in (4).

$$f_t = \sigma[W_f \cdot (h_{t-1}, z_t) + b_f] \quad (4)$$

Input gate takes the output of the previous state h_{t-1} and current input z_t, and then passes these values through the sigmoid function as well as tanh function as shown in (5) and (6). The input gate decides how much the input value flows into the memory cells.

$$i_t = \sigma[W_i \cdot (h_{t-1}, z_t) + b_i] \quad (5)$$

$$\tilde{C}_t = \tanh[W_c \cdot (h_{t-1}, z_t) + b_c] \quad (6)$$

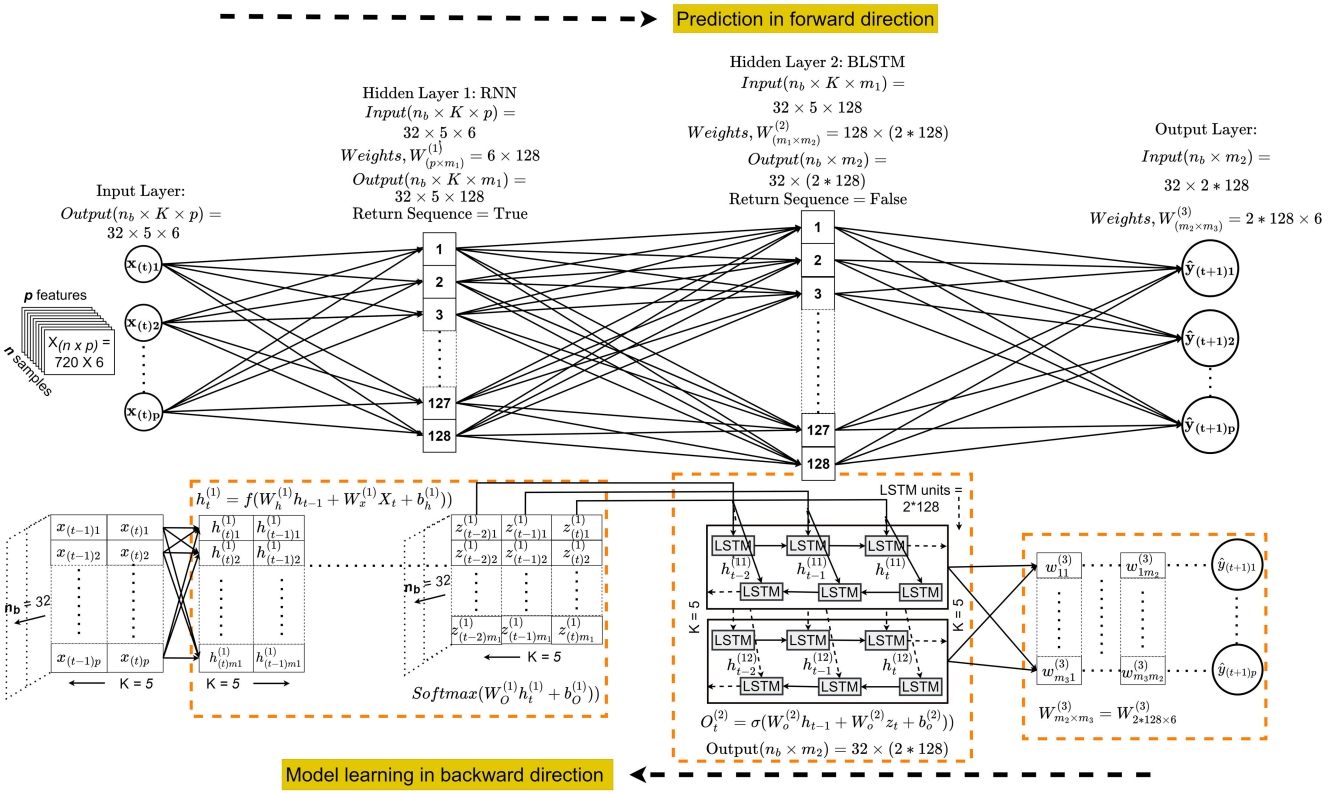

FIGURE 7. Representation of the proposed neural network architecture which contains single-layered RNN with 128 hidden units and single-layered BLSTM with 128 hidden units.

At this point, the previous cell state $\tilde{C}_t$ is updated to the new cell state C_t as shown in (7).

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{7}$$

The output gate determines which output (for the next hidden state) to generate from the current cell state. It considers the output of the previous state h_{t-1} and current input z_t, and passes these values through the sigmoid function as shown in (8), and passes updated state C_t to tanh function and update the output of the current hidden layer as shown in (9). Output is the multiplication of the Sigmoid function and the tanh function.

$$O_t = \sigma[W_o \cdot (h_{t-1}, z_t) + b_o] \tag{8}$$

$$h_t = o_t \odot \tanh(C_t) \tag{9}$$

where $\odot$ is Hadamard's (element-wise) product. The new cell state and the new hidden unit output are then carried over to the next time step.

4) OUTPUT DENSE LAYER

The return sequence = False in the BLSTM layer shows that it will only return the last hidden state of the BLSTM layer. The BLSTM layer is followed by the dense output layer of shape $(2 * 128) \times 6$, which gives the desired predicted value $\tilde{Y}_{t+1}$.

F. PREDICTION MODEL TRAINING

The RNN-BLSTM model has trained over 500 epochs, taking 134 msec per epoch (8 msec per step). As described in the previous section, the single-layered RNN consists of 128 hidden units, whereas the single-layered BLSTM also comprises 128 hidden units. Fig. 8 depicts the training and the validation loss over the number of epochs. It can be observed that the model performance is improving with experience. Mean square error loss shows that the model perfectly fits over the training data as well as over the validation data. The neural network contains 223, 557 parameters without any non-trainable parameters. Furthermore, the summary of the neural network is presented in Table 3.

G. QoS-AWARE PRIORITY-BASED DYNAMIC NETWORK RESOURCES ALLOCATION

Network resources can be dedicated or shared. The dedicated network resource model does not allow the network to change its resources for a QoS class during congestion, especially when the demand metric of each QoS class changes dynamically. In other words, this dedicated scheme has low scalability and flexibility. On the other hand, the shared network resource model allows the network to dynamically change the resources for each QoS class. As the network resources are limited, the dedicated allocation of excessive network resources to a QoS class can degrade the performance of other QoS classes. This paper uses a

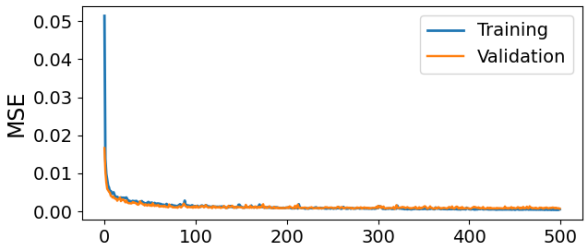

FIGURE 8. Training and the validation loss of the proposed prediction model vs number of Epochs.

TABLE 3. Summary of the RNN-BLSTM model.

Layer (type)	Output Shape	Param #
simple_rnn (SimpleRNN)	(None, 5, 128)	17280
dropout (Dropout)	(None, 5, 128)	0
bidirectional (Bidirectional)	(None, 256)	196608
flatten (Flatten)	(None, 256)	0
dense_1 (Dense)	(None, 6)	1542
activation (Activation)	(None, 6)	0
Total params: 223,557 Trainable params: 223,557 Non-trainable params: 0		

QoS-aware dynamic network resources allocation scheme that considers the predicted traffic in each QoS class along with its priority [73]. The priority of each QoS class is set based on the two QoS parameters; 1) packet loss and 2) bandwidth requirement, which is assumed to be the same for similar applications. The network management protocol traffic has the highest priority, whereas the unidentified traffic has the lowest priority.

The dynamic network resources allocation scheme is shown in Algorithm 2. Let $\hat{\boldsymbol{y}}_{(t+\tau)\boldsymbol{p}}$ represents the predicted traffic in p QoS class for time $(t+\tau)$, predicted at time t (where $\tau = 5$). Assume that j_p denotes the priority vector having values $j_p = [6, 5, 4, 3, 2, 1]$ for the six considered QoS classes, respectively. Then, the prioritized bandwidth request for each QoS class p can be represented as $\gamma_{(t+\tau)p}$, which can be expressed by:

$$\gamma_{(t+\tau)p} = j_p \cdot y_{(t+\tau)p} \tag{10}$$

The above equation calculates a simple prioritized bandwidth demand of each QoS class. But, the allocation of resources to a QoS class can affect the other QoS classes. Thus, (11) calculates a fair bandwidth ratio $\beta_{(t+\tau)p}$ which can be requested by each QoS class p for the next time stamp $(t+\tau)$:

$$\beta_{(t+\tau)p} = \frac{\gamma_{(t+\tau)p}}{\sum_{n=1}^{6} \gamma_{(t+\tau)n}}, \quad \beta_{(t+\tau)p} \leq \mu_p^{max} \tag{11}$$

where μ_p^{max} is the maximum limit to allocate the network resources to the QoS classes from the shared resource pool. The nominator shows the prioritized bandwidth demand of a QoS class, whereas the denominator shows the prioritized bandwidth demand of all the QoS classes.

Finally, the network will take the decision about the amount of network resources that can be allocated to each QoS class, based on the prioritized bandwidth request of each QoS class $\beta_{(t+\tau)p}$ and the available network resources in terms of bandwidth B_{total}:

$$B_p^{max}(t+\tau) = B_{total} \times \mu_p^{min} + B_{remaining} \times \beta_{(t+\tau)p} \tag{12}$$

In the above equation, μ_p^{min} is the minimum guaranteed bandwidth that must be allocated to each QoS class. $B_{remaining}$ is the remaining bandwidth that can be shared among all the QoS classes. B_p^{max} is the maximum bandwidth that can be assigned to each QoS class so that the allocated bandwidth should not exceed the total available network bandwidth. Let's assume that the length of the priority vector j_p is represented by m and the number of iterations of the while loop is n, the time complexity of Algorithm 2 can be determined as $\mathcal{O}(n \times m)$, resulting to $\mathcal{O}(n^2)$.

Algorithm 2 QoS-aware Resource Allocation Algorithm:

Output: Assign network resources to each QoS class for the next 5 minutes based on the predicted traffic and the priority of the QoS class, $B_p^{max}(t+\tau)$
Input : μ_p^{min}, μ_p^{max}, j_p, $y_{(t+\tau)p}$
Initialization : $B_p^{max}(t)$
while *Network Resources Allocation is active* **do**
- Calculate the prioritized bandwidth request for each QoS class, $\gamma_{(t+\tau)p}$ using
  - predicted traffic $y_{(t+\tau)p}$
  - priority vector j_p
- Calculate the required minimum network resources for each QoS class, μ_p^{min}
- Calculate the required shared bandwidth ratio for each QoS class, $\beta_{(t+\tau)p}$

if $\beta_{(t+\tau)p} > \mu_p^{max}$ **then**
Assign $\beta_{(t+\tau)p}$ to the QoS class with maximum bandwidth μ_p^{max}
end
Assign network resources $B_p^{max}(t+\tau)$ for $(t+\tau)$ time
end

V. EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION METRICS

This section provides the experimental setup for the simulation, an in-depth description of the evaluation metrics, and a description of the prediction model training.

A. EXPERIMENTAL SETUP

The setup of our examination is described in detail in this section. The network traffic is captured through the campus network consisting of three building blocks, including a data center as shown in Fig. 2. The machine that captures the network traffic is directly connected through a LAN port to the switch model Foundry Fast Iron 1600X. The machine used to capture the network traffic for the purpose of dataset

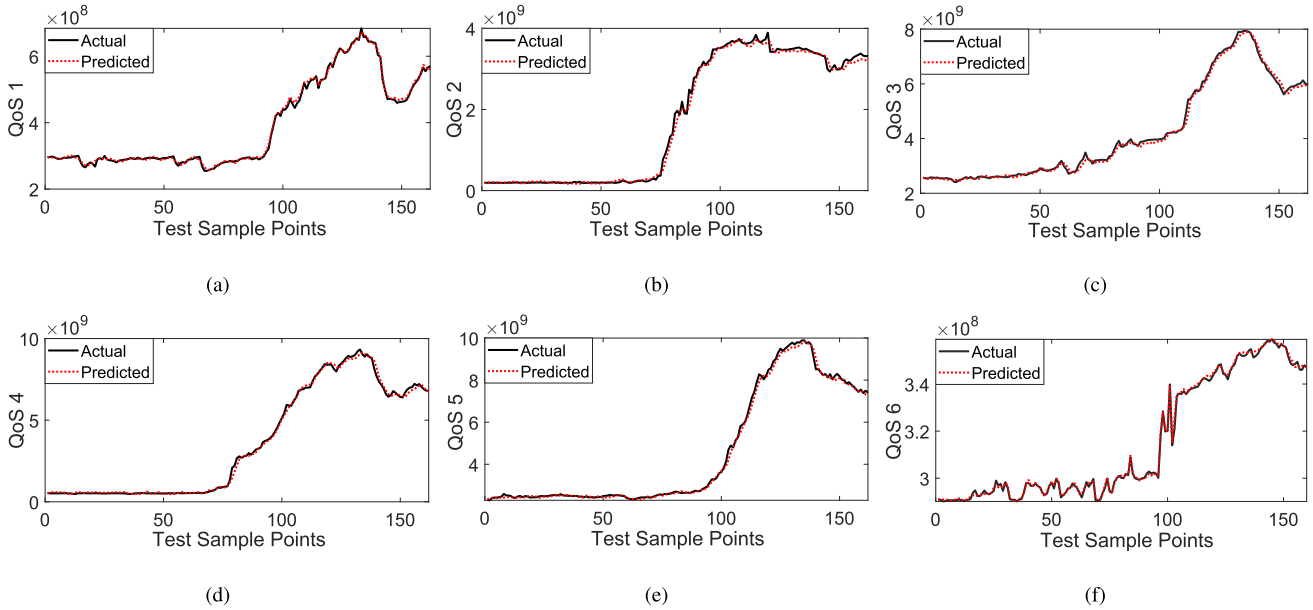

FIGURE 9. Prediction of (a) Network management traffic (b) Live audio/video traffic (c) HTTP and live streaming and video traffic (d) Fun applications and online gaming traffic (e) Downloading and system update traffic (f) Unidentified traffic.

collection is a 7th generation OptiPlex 7060 having 16 GB DDR4 and processing power up to 3.20 GHz accommodating 6 cores and 12 threads.

In the first step, the network traffic is identified through a DPI engine, which forwards these results to the MySQL database through an API that updates the flow level entries inside the MySQL database after every 10 seconds. More details about the database is can be found in Sec. IV-C1.

This has resulted in a total of 720 data samples corresponding to 48 hours of total monitoring time. For training purposes, we divide the dataset into two parts. The dataset's first 560 samples (78%) are used for training purposes. The remaining 160 samples (22%) are used to evaluate the effectiveness of the proposed method and are referred to as the test dataset.

B. PERFORMANCE EVALUATION METRICS

In this paper, we are using three performance metrics to evaluate the proposed prediction algorithm, which are the following:

1) MEAN ABSOLUTE ERROR (MAE)

It is used to analyze the average inaccuracy expected in the predicted time series. Smaller value shows that the predicted values are comparable with the actual values.

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{13}$$

2) MEAN ABSOLUTE PERCENTAGE ERROR (MAPE)

It is the proportion of the average absolute difference between the actual and the predicted value divided by the actual value. This metric can be used in a dataset which is free of zeros and extreme values.

$$\text{MAPE} = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \tag{14}$$

3) ROOT MEAN SQUARE ERROR (RMSE)

It defines the square root of the mean absolute error. RMSE can be compared with MAE to observe the inaccuracies in the time series forecast. The more the gap between the MAE and RMSE, the more erratic will be the error size. RMSE penalizes the outliers or extreme errors more than the small errors.

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \tag{15}$$

VI. PERFORMANCE EVALUATION AND RESULTS

In this section, we analyse the degree to which the generated predictions are similar to the actual measurements by presenting the assessments acquired using the suggested framework. More specifically, we examine the prediction accuracy and the error loss of the proposed RNN-BLSTM approach using the evaluation metrics discussed in Section V-B. Also, we compare our approach with other competitive approaches, such as ARIMA, SVM and LSTM, in terms of accuracy, time and energy consumption. Finally, we present our analysis of the dynamic allocation of network resources using the predicted traffic and the priority metric of each QoS class.

A. PREDICTION PERFORMANCE OF RNN-BLSTM

The multivariate time series prediction results for the considered QoS classes are presented in Fig. 9. The black

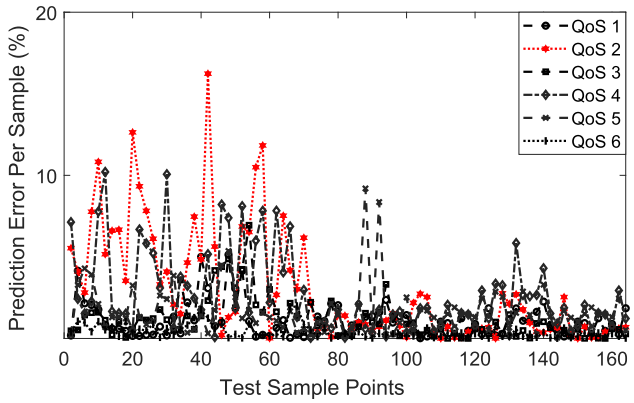

FIGURE 10. Per-sample prediction error on test dataset for all the QoS classes.

TABLE 4. Error metric.

	MAE (MB)	MAPE (%)	RMSE (MB)
QoS 1	7.33	1.84	10.39
QoS 2	53.88	4.06	62.30
QoS 3	80.33	1.90	114.70
QoS 4	90.68	3.07	105.30
QoS 5	99.10	2.43	88.40
QoS 6	1.80	0.59	2.86

solid line curve shows the actual time series, whereas the red dotted line curve shows the predicted traffic. Fig. 9(a) compares the actual and the predicted network management traffic inside the total network traffic. Fig. 9(b) represents the actual and the predicted live audio/video calling traffic inside total network traffic. Fig. 9(c) shows the actual and the predicted HTTP traffic, including live streaming and video traffic inside total internet traffic. Fig. 9(d) compares the actual and the predicted fun applications traffic as well as the online gaming traffic inside the total network traffic. Fig. 9(e) shows the actual and the predicted downloading and system update protocol traffic, and Fig. 9(f) shows the actual and the predicted traffic of QoS class 6 that includes unidentified traffic.

B. PER-SAMPLE PREDICTION ERROR OF RNN-BLSTM

We use the MAPE formula presented in Eq. 14 to analyze the per-sample prediction performance without averaging it over the number of test samples. Fig. 10 shows that the per-sample prediction error is less than 17% for all the test samples. Moreover, 94% of the test samples have a prediction error of less than 2%. The highest error per sample is observed for QoS class 2 at test sample point 21, which is 17%. Further, it is also evident from Fig. 10 that the prediction performance improves with the learning experience, and the error spikes gradually fade.

C. PREDICTION ERROR OF RNN-BLSTM

The proposed model is evaluated based on the performance metrics presented in Sec. V-B. Table 4 presents the MAE, MAPE, and RMSE analysis on the test datasets. The highest MAPE value is observed for QoS class 2, whereas QoS class 6 outperforms all the other QoS classes regarding prediction accuracy. MAE and RMSE show that both the values are closer to each other and are not diverging from each other, so the error is less erratic. This analysis shows that the proposed framework can be used to predict the multivariate time series network traffic with high accuracy.

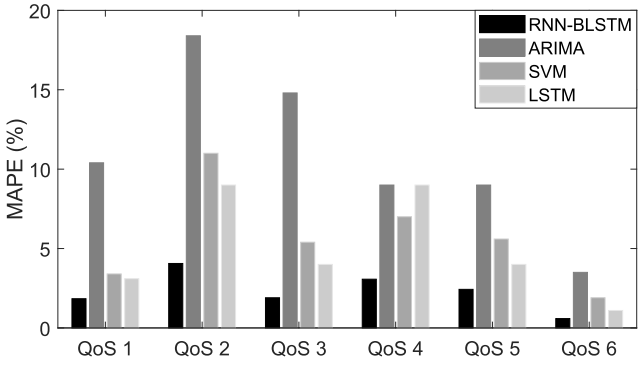

FIGURE 11. MAPE for RNN-BLSTM, ARIMA, SVM and LSTM within each QoS class.

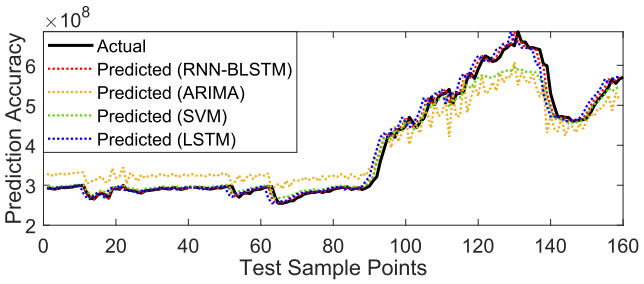

FIGURE 12. Prediction of QoS 1 with RNN-BLSTM, ARIMA, SVM and LSTM.

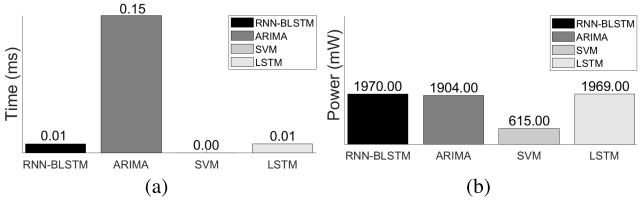

FIGURE 13. (a) Latency of the considered prediction algorithms, (b) Power consumption by the prediction algorithms.

D. PERFORMANCE COMPARISON OF RNN-BLSTM WITH OTHER PREDICTION MODELS

In this sub-section, we compare the prediction accuracy of the proposed framework with the statistical and machine learning model. For this purpose, we use ARIMA(4,1,0), SVM (radial basis function), and LSTM (single-layered with 128 units) neural network models. The RMSE comparison of the aforementioned prediction techniques is presented in Fig. 11. It is evident from the graph that the proposed RNN-BLSTM model performs well compared to the other prediction algorithms. It can be observed from Fig. 12 that the LSTM, ARIMA, and SVM follow the actual trend, but the prediction accuracy is less.

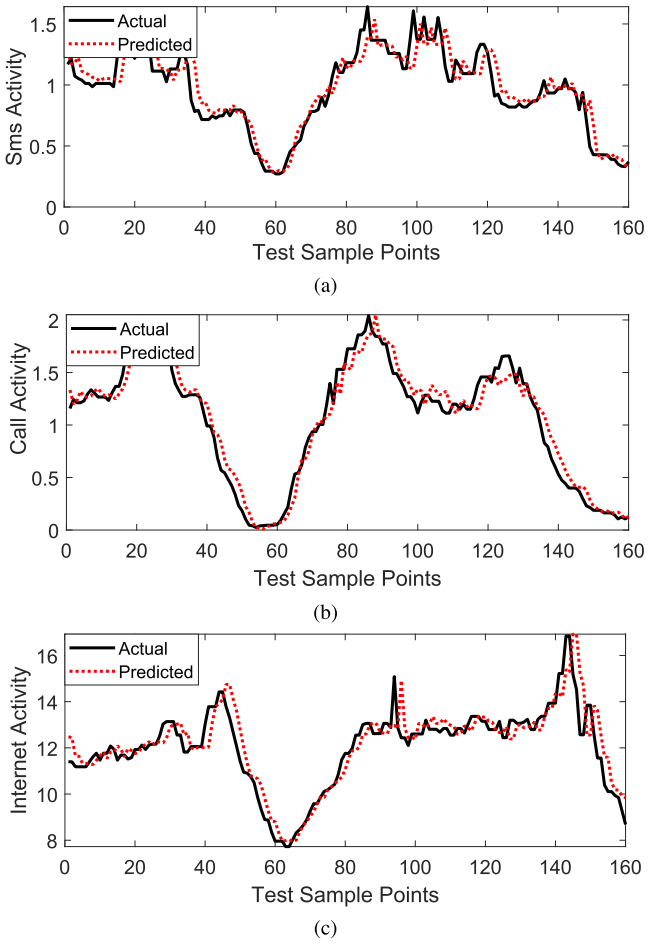

FIGURE 14. Predbiction of (a) SMS activity, (b) Call activity, and (c) Internet activity within the base station 8028.

TABLE 5. Error metric for Telecom Italia Dataset.

Traffic Type	MAE (MB)	MAPE (%)	RMSE (MB)
SMS	0.09	8.60	0.13
Calls	0.09	16.38	0.12
Internet	0.54	4.20	0.78

E. TIME AND ENERGY CONSUMPTION OF RNN-BLSTM

The prediction algorithms were evaluated based on prediction accuracy in the previous sub-section. In this sub-section, we compare the prediction algorithms based on the prediction time and total power consumption. For the memory calculation, we used the "top" command at Raspberry Pi, where we ran the memory and power consumption calculations different prediction approaches. Additionally, for power consumption measurement, we used the "powertop" tool.

Fig. 13(a) displays the time required by each approach for a single prediction. It is observed that SVM is the quickest approach for prediction, closely followed by RNN-BLSTM, LSTM and ARIMA.

Fig. 13(b) depicts the total power consumption of each approach, timed for 20 seconds (from training to predicting),

TABLE 6. Prediction accuracy comparison with the literature.

Reference	Evaluation Metric	Univariate Dataset	Multivariate Dataset	Result
[54]	MAPE	✓		29.46 %
[55]	MAPE	✓		9.1 %
[56]	RMSE	✓		0.6199
[75]	MAPE			13.40 %
[57]	MAPE		✓	11 %
Our Work (RNN-BLSTM)	MAPE	✓		SMS = 8.6 % Calls = 16.38 % Internet = 4.2 %
Our Work (RNN-BLSTM)	MAPE		✓	9.73 %
Our Work (RNN-BLSTM)	RMSE	✓		SMS = 0.13 Calls = 0.12 Internet = 0.78
Our Work (RNN-BLSTM)	RMSE		✓	0.34

and it is observed that the RNN-BLSTM approach requires the most power, followed by LSTM, ARIMA and SVM.

F. VALIDATION OF THE PROPOSED PREDICTION MODEL

Validation of the proposed framework is done on the Telecom Italia dataset [53]. Specifically, cell number 8028 is chosen, and the data is collected from 01−11−2013 to 07−11−2013. Moving the median over the window size of 3 is applied to remove the spikes and to keep the randomness in the time series. Moreover, the considered dataset contains the outgoing SMS, the outgoing call, and the internet activity. The same prediction model discussed in Sec. IV-E is applied with the same training parameters, except for the shape of the input and the output layer (3 in the considered scenario). Results in Fig. 14 show the effectiveness of the proposed prediction model having better accuracy than the prediction models proposed in the literature, e.g., [74] (MAPE value of 16.62% for univariate traffic). Since Our considered dataset contains considerable randomness, which minimizes the seasonality in the time series. The error metric is shown in Table 5.

Table 6 compares the proposed prediction model with the existing literature. Based on how closely the dataset and cellular traffic pattern resemble the considered dataset [53], a comparison is made. The suggested RNN-BLSTM performs better than the other techniques for predicting univariate time series internet traffic. Our suggested methodology still outperforms the frameworks addressed in the literature when we use multivariate time series forecasting to predict calls, text messages, and internet traffic, all at once.

G. QoS-AWARE RESOURCE ALLOCATION

In this sub-section, we present the results which are obtained using the Algorithm 2 described in Sec. IV-G. The priority is set in a way that the maximum possible resources will be allocated to the QoS class, which is sensitive to bandwidth and packet loss. Fig. 15 presents the results when the available network resources are shared among the different QoS classes based on the priority metric mentioned in the Sec. IV-G. The green dotted line shows the link's maximum capacity. In contrast, the solid black line represents the total

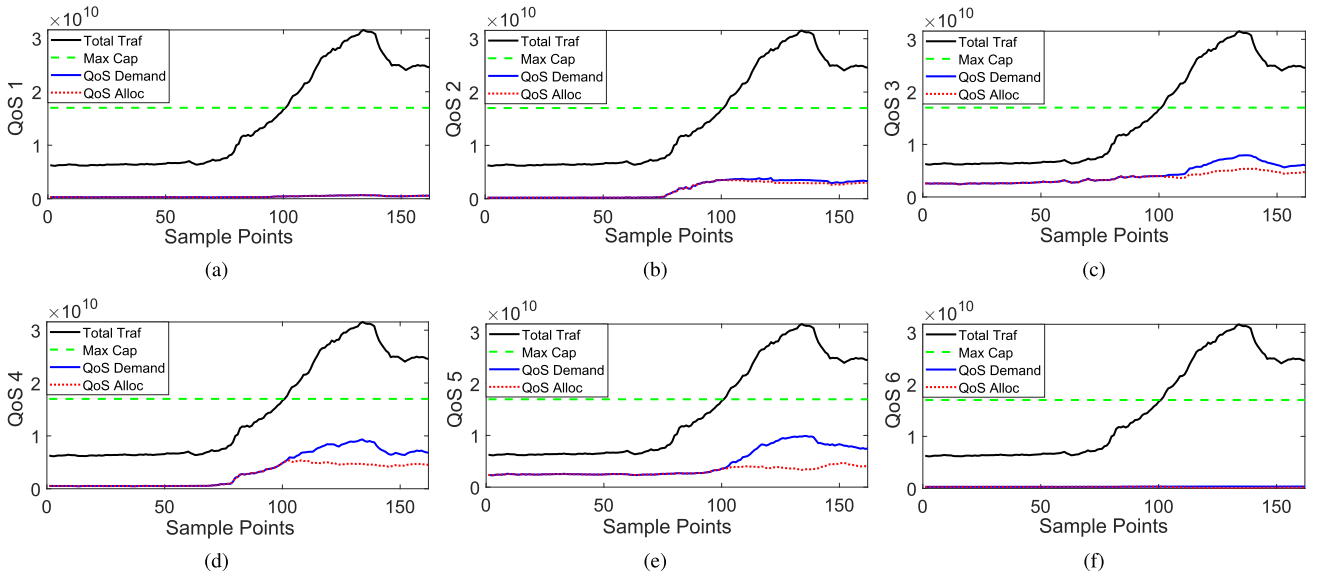

FIGURE 15. Bandwidth allocation of (a) Network management traffic (b) Live audio/video traffic (c) HTTP and live streaming and video traffic (d) Fun applications and online gaming traffic (e) Downloading and system update traffic (f) Unidentified traffic, during congestion in the network.

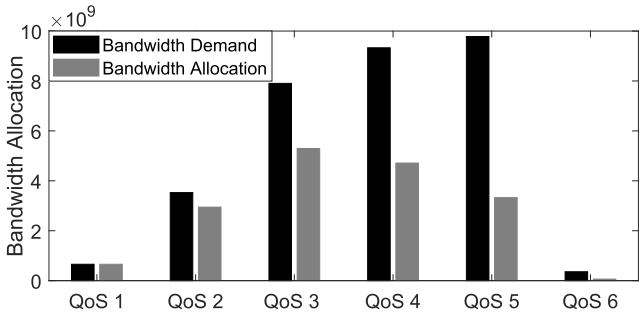

FIGURE 16. Allocation of network resources based on the predicted traffic and the priority of the QoS class during peak congestion time.

network traffic passing through the considered link at a time $(t + \tau)$. The blue line shows the bandwidth demand of each QoS class, and the red dotted line represents the allocated bandwidth to meet the requested demand of that QoS class. Fig. 15(a) depicts the QoS class 1, which contains the network management traffic protocols. The network will allocate the maximum requested bandwidth to QoS 1 using μ_p^{min} = total traffic inside QoS class 1. The remaining bandwidth will be shared among the other QoS classes where the minimum threshold bandwidth μ_p^{min} is set to 20% of the requested bandwidth. The allocated resources (red dotted line) during the congestion can be analysed in Fig. 15. QoS class 2 shows the live audio video calling, which is sensitive to delay and packet loss. During congestion, a maximum of 90% of the required bandwidth $B_2^{max} = 0.9 \times y_{(t+\tau)2}$ will be allocated to QoS class 2 as shown in Fig. 15(b). The QoS Class 3 represents the HTTP browsing and video streaming traffic, whereas QoS class 4 denotes the social networking applications and online gaming. Both these classes require high bandwidth but they are less delay-sensitive than QoS classes 1 and 2. As a result, QoS class 3 and 4 will get 85% of the maximum demanded bandwidth, that is $B_{3,4}^{max} = 0.85 \times y_{(t+\tau)3,4}$, which can be analysed in Fig. 15(c) and 15(d). QoS class 5 contains downloading and system update traffic, and QoS 6 contains unidentified traffic information. As both QoS classes are not sensitive to delay and packet loss, a maximum of 70% of the required bandwidth will be allocated to the QoS class 5 and 6 which is equivalent to $B_{5,6}^{max} = 0.70 \times y_{(t+\tau)5,6}$ as depicted in Fig. 15(e) and 15(f). The bandwidth allocation can be adjusted depending on the scenario. In this manner, the network can accommodate excess traffic that exceeds the maximum link capacity.

In the scenario, we assume that the maximum link capacity is 56.6 Mbps ($1.7e^{10}$ bits / 5 minutes). The network will operate normally if bandwidth demands do not exceed the maximum link capacity. The network congestion can be observed at test sample 101 when the network traffic demand (black line) surpasses available link capacity (green dotted line).

The most challenging case of resource allocation is observed for the test sample 134, and is presented in Fig. 16. During this time, the total demand of QoS classes is approximately 105 Mbps, whereas the total link capacity is 56.6 Mbps. It can be seen that the network traffic remains unaffected during the congestion. In contrast, the downloading and system update traffic is mostly affected and is not sensitive to bandwidth and packet loss.

VII. CONCLUSION AND FUTURE WORK

This paper proposes a framework to identify, classify and predict the heterogeneous network traffic using RNN-BLSTM neural network. We have predicted the network traffic in each QoS class for better network resource utilization for the case of dynamic link allocation based on traffic requirements in each QoS class. The average prediction accuracy of 97.68% is

achieved using RNN-BLSTM neural network for multivariate QoS classes, which outperforms the existing prediction work on the multivariate time series datasets. The predicted traffic allocates the network resources proactively to avoid future link congestion for sensitive internet traffic. The results show that the dynamic network resources allocation framework provides more bandwidth to sensitive traffic than the fair bandwidth allocation scheme. In our future studies, we will validate the proposed framework using network simulators.

REFERENCES

[1] U. Cisco, "Cisco annual internet report (2018–2023) white paper," Cisco, San Jose, CA, USA, White Paper C11, 2020, vol. 10, no. 1, pp. 1–35. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[2] N. H. Mahmood et al., "White paper on critical and massive machine type communication towards 6G," 2020, *arXiv:2004.14146*.

[3] Y. Guo, Q. Duan, and S. Wang, "Service orchestration for integrating edge computing and 5G network: State of the art and challenges," in *Proc. IEEE World Congr. Services (SERVICES)*, Oct. 2020, pp. 55–60.

[4] H. Chergui, A. Ksentini, L. Blanco, and C. Verikoukis, "Toward zero-touch management and orchestration of massive deployment of network slices in 6G," *IEEE Wireless Commun.*, vol. 29, no. 1, pp. 86–93, Feb. 2022.

[5] J. Kaur, M. A. Khan, M. Iftikhar, M. Imran, and Q. E. Ul Haq, "Machine learning techniques for 5G and beyond," *IEEE Access*, vol. 9, pp. 23472–23488, 2021.

[6] *Zero-Touch Network and Service Management (ZSM); Reference Architecture*, document ETSI GS ZSM 002, Group Specification (GS) ETSI GS ZSM, 2019.

[7] *Experiential Networked Intelligence (ENI) Requirements*, document ENI, version 2.1. 1, ETSI GS ETSI, Sophia Antipolis, France, 2019.

[8] K. Velasquez, D. P. Abreu, M. Curado, and E. Monteiro, "Resource orchestration in 5G and beyond: Challenges and opportunities," *Comput. Commun.*, vol. 192, pp. 311–315, Aug. 2022.

[9] *Distributed Artificial Intelligence-Driven Open & Programmable Architecture for 6G Networks*, ADROIT6G, Athens, Greece, 2023.

[10] J. Zhang and N. Ansari, "On assuring end-to-end QoE in next generation networks: Challenges and a possible solution," *IEEE Commun. Mag.*, vol. 49, no. 7, pp. 185–191, Jul. 2011.

[11] S. Malisuwan, D. Milindavanij, and W. Kaewphanuekrungsi, "Quality of service (QoS) and quality of experience (QoE) of the 4G LTE perspective," *Int. J. Future Comput. Commun.*, vol. 5, no. 3, pp. 158–162, 2016.

[12] Z. Xu, F. Yut, C. Liu, and X. Chen, "ReForm: Static and dynamic resource-aware DNN reconfiguration framework for mobile device," in *Proc. 56th ACM/IEEE Design Autom. Conf. (DAC)*, Jun. 2019, pp. 1–6.

[13] D. Warneke and O. Kao, "Exploiting dynamic resource allocation for efficient parallel data processing in the cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 6, pp. 985–997, Jun. 2011.

[14] J. Zhang, F. Li, and F. Ye, "Network traffic clustering with QoS-awareness," *China Commun.*, vol. 19, no. 3, pp. 202–214, Mar. 2022.

[15] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Müller, and K. Hanssgen, "A survey of payload-based traffic classification approaches," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 2, pp. 1135–1156, 2nd Quart., 2014.

[16] R. M. AlZoman and M. J. F. Alenazi, "A comparative study of traffic classification techniques for smart city networks," *Sensors*, vol. 21, no. 14, p. 4677, Jul. 2021.

[17] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. M. Choudhury, and A. K. Qin, "A survey on modern deep neural network for traffic prediction: Trends, methods and challenges," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 4, pp. 1544–1561, Apr. 2022.

[18] A.-K. Al-Tamimi, R. Jain, and C. So-In, "Dynamic resource allocation based on online traffic prediction for video streams," in *Proc. IEEE 4th Int. Conf. Internet Multimedia Services Archit. Appl.*, Dec. 2010, pp. 1–6.

[19] M. M. Tajiki, B. Akbari, and N. Mokari, "QRTP: QoS-aware resource reallocation based on traffic prediction in software defined cloud networks," in *Proc. 8th Int. Symp. Telecommun. (IST)*, Sep. 2016, pp. 527–532.

[20] A. Akçapinar, Ö. Gürer, and V. Rodoplu, "ARIMA-based traffic forecasting for quality of service (QoS) flow routing in sixth generation (6G) networks," in *Proc. Innov. Intell. Syst. Appl. Conf. (ASYU)*, Sep. 2022, pp. 1–5.

[21] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent neural networks for time series forecasting: Current status and future directions," *Int. J. Forecasting*, vol. 37, no. 1, pp. 388–427, Jan. 2021.

[22] S. T. Aung and T. Thein, "Internet traffic categories demand prediction to support dynamic QoS," in *Proc. 5th Int. Conf. Comput. Commun. Syst. (ICCCS)*, May 2020, pp. 650–654.

[23] W. Jiang, "Cellular traffic prediction with machine learning: A survey," *Expert Syst. Appl.*, vol. 201, Sep. 2022, Art. no. 117163.

[24] W. A. Aziz, H. K. Qureshi, A. Iqbal, and M. Lestas, "Accurate prediction of streaming video traffic in TCP/IP networks using DPI and deep learning," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, Jun. 2020, pp. 310–315.

[25] R. D. Alamsyah and S. Suyanto, "Speech gender classification using bidirectional long short term memory," in *Proc. 3rd Int. Seminar Res. Inf. Technol. Intell. Syst. (ISRITI)*, Dec. 2020, pp. 646–649.

[26] S. Xue, Z. Yan, Z. Huang, and L. Dai, "Rapid speaker adaptation based on d-code extracted from BLSTM-RNN in LVCSR," in *Proc. 10th Int. Symp. Chin. Spoken Lang. Process. (ISCSLP)*, Oct. 2016, pp. 1–5.

[27] J. Zhang, J. Tang, and L.-R. Dai, "RNN-BLSTM based multi-pitch estimation," in *Proc. Interspeech*, Sep. 2016, pp. 1785–1789.

[28] A. Tamamori, T. Hayashi, T. Toda, and K. Takeda, "An investigation of recurrent neural network for daily activity recognition using multi-modal signals," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Dec. 2017, pp. 1334–1340.

[29] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "A comparison of ARIMA and LSTM in forecasting time series," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 1394–1401.

[30] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the ARIMA model," in *Proc. 16th Int. Conf. Comput. Modeling Simulation*, Mar. 2014, pp. 106–112.

[31] *Understanding ARIMA Models for Machine Learning*, Capital One, McLean, VA, USA, 2021.

[32] J. Fattah, L. Ezzine, Z. Aman, H. E. Moussami, and A. Lachhab, "Forecasting of demand using ARIMA model," *Int. J. Eng. Bus. Manage.*, vol. 10, Oct. 2018, Art. no. 1847979018808673.

[33] H. Z. Moayedi and M. A. Masnadi-Shirazi, "ARIMA model for network traffic prediction and anomaly detection," in *Proc. Int. Symp. Inf. Technol.*, 2008, pp. 1–6.

[34] R. Nau, "Introduction to ARIMA models," Fuqua School Bus., Duke Univ., Durham, NC, USA, 2014.

[35] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[36] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Phys. D, Nonlinear Phenomena*, vol. 404, Mar. 2020, Art. no. 132306.

[37] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput.*, vol. 1, no. 2, pp. 270–280, Jun. 1989.

[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[39] S. Hochreiter and J. Schmidhuber, "LSTM can solve hard long time lag problems," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 9, 1996, pp. 1–8.

[40] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 855–868, May 2009.

[41] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, Jul. 2019.

[42] S. K. U. Zaman, A. I. Jehangiri, T. Maqsood, A. I. Umar, M. A. Khan, N. Z. Jhanjhi, M. Shorfuzzaman, and M. Masud, "COME-UP: Computation offloading in mobile edge computing with LSTM based user direction prediction," *Appl. Sci.*, vol. 12, no. 7, p. 3312, Mar. 2022.

[43] S. Barmpounakis, L. Magoula, N. Koursioumpas, R. Khalili, J. M. Perdomo, and R. P. Manjunath, "LSTM-based QoS prediction for 5G-enabled connected and automated mobility applications," in *Proc. IEEE 4th 5G World Forum (5GWF)*, Oct. 2021, pp. 436–440.

[44] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul./Aug. 2005, pp. 2047–2052.
[45] G. Du, Z. Wang, B. Gao, S. Mumtaz, K. M. Abualnaja, and C. Du, "A convolution bidirectional long short-term memory neural network for driver emotion recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4570–4578, Jul. 2021.
[46] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *J. Mach. Learn. Res.*, vol. 3, no. 1, pp. 115–143, 2003.
[47] B. K. Reddy and D. Delen, "Predicting hospital readmission for lupus patients: An RNN-LSTM-based deep-learning methodology," *Comput. Biol. Med.*, vol. 101, pp. 199–209, Oct. 2018.
[48] G. Bian, J. Liu, and W. Lin, "Internet traffic forecasting using boosting LSTM method," *DEStech Trans. Comput. Sci. Eng.*, Jan. 2018.
[49] W. Jiang, "Internet traffic matrix prediction with convolutional LSTM neural network," *Internet Technol. Lett.*, vol. 5, no. 2, p. e322, Mar. 2022.
[50] W. Shihao, Z. Qinzheng, Y. Han, L. Qian-mu, and Q. Yong, "A network traffic prediction method based on LSTM," *ZTE Commun.*, vol. 17, pp. 19–25, Jun. 2019.
[51] J. Chen, H. Xing, H. Yang, and L. Xu, "Network traffic prediction based on LSTM networks with genetic algorithm," in *Signal and Information Processing, Networking and Computers* (Lecture Notes in Electrical Engineering), vol. 550. New York, NY, USA: Springer-Verlag, 2019, pp. 411–419.
[52] P. Bermolen and D. Rossi, "Support vector regression for link load prediction," *Comput. Netw.*, vol. 53, no. 2, pp. 191–201, Feb. 2009.
[53] G. Barlacchi, M. De Nadai, R. Larcher, A. Casella, C. Chitic, G. Torrisi, F. Antonelli, A. Vespignani, A. Pentland, and B. Lepri, "A multi-source dataset of urban life in the city of Milan and the Province of Trentino," *Sci. Data*, vol. 2, no. 1, pp. 1–15, Oct. 2015.
[54] X. Chen, G. Chuai, K. Zhang, and W. Gao, "Spatial–temporal cellular traffic prediction: A novel method based on causality and graph attention network," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2023, pp. 1–6.
[55] W. Jiewu, F. Wentao, H. Chunjing, and Z. Xing, "User traffic collection and prediction in cellular networks: Architecture, platform and case study," in *Proc. 4th IEEE Int. Conf. Netw. Infrastruct. Digit. Content*, Sep. 2014, pp. 414–419.
[56] X. Xu, S. Gao, and Z. Jiang, "LSTCN: An attention-based deep neural network model combining LSTM and TCN for cellular network traffic prediction," in *Proc. 5th Int. Conf. Commun. Inf. Syst. (ICCIS)*, Oct. 2021, pp. 34–38.
[57] M. Nakip, B. C. Gül, V. Rodoplu, and C. Gözeliş, "Predictability of Internet of Things traffic at the medium access control layer against information-theoretic bounds," *IEEE Access*, vol. 10, pp. 55602–55615, 2022.
[58] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Applying deep learning approaches for network traffic prediction," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2017, pp. 2353–2358.
[59] Z. Zhou, X. Chen, H. Liao, Z. Gan, F. Xiao, Q. Tu, W. Sun, Y. Liu, S. Mumtaz, and M. Guizani, "Collaborative learning-based network resource scheduling and route management for multi-mode green IoT," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 2, pp. 928–939, Jun. 2023.
[60] S. Tomovic and I. Radusinovic, "A new prediction based technique for quality of service aware traffic engineering," in *Proc. 27th Telecommun. Forum (TELFOR)*, Nov. 2019, pp. 1–4.
[61] Q. Zhu, D. Li, Y. Xin, X. Yu, and G. Mu, "A survey on network traffic identification," in *Proc. Int. Conf. Artif. Intell. Secur.* Cham, Switzerland: Springer, 2019, pp. 91–100.
[62] A. Callado, C. Kamienski, G. Szabo, B. P. Gero, J. Kelner, S. Fernandes, and D. Sadok, "A survey on Internet traffic identification," *IEEE Commun. Surveys Tuts.*, vol. 11, no. 3, pp. 37–52, 3rd Quart., 2009.
[63] I. Lohrasbinasab, A. Shahraki, A. Taherkordi, and A. Delia Jurcut, "From statistical-to machine learning-based network traffic prediction," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 4, p. e4394, 2022.
[64] S. Patel, A. Gupta, Nikhil, S. Kumari, M. Singh, and V. Sharma, "Network traffic classification analysis using machine learning algorithms," in *Proc. Int. Conf. Adv. Comput., Commun. Control Netw. (ICACCCN)*, Oct. 2018, pp. 1182–1187.
[65] B. Yang and D. Liu, "Research on network traffic identification based on machine learning and deep packet inspection," in *Proc. IEEE 3rd Inf. Technol., Netw., Electron. Autom. Control Conf. (ITNEC)*, Mar. 2019, pp. 1887–1891.
[66] R. T. El-Maghraby, N. M. A. Elazim, and A. M. Bahaa-Eldin, "A survey on deep packet inspection," in *Proc. 12th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2017, pp. 188–197.
[67] A. Gustavo, C. Binnig, I. Pandis, K. Salem, J. Skrzypczak, R. Stutsman, L. Thostrup, T. Wang, Z. Wang, and T. Ziegler, "DPI: The data processing interface for modern networks," in *Proc. CIDR*, 2019, pp. 1–7.
[68] L. Deri, M. Martinelli, T. Bujlow, and A. Cardigliano, "NDPI: Open-source high-speed deep packet inspection," in *Proc. Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Aug. 2014, pp. 617–622.
[69] J. Biswas and A. Ashutosh, "An insight in to network traffic analysis using packet sniffer," *Int. J. Comput. Appl.*, vol. 94, no. 11, pp. 39–44, May 2014.
[70] C. Yu, J. Lan, J. Xie, and Y. Hu, "QoS-aware traffic classification architecture using machine learning and deep packet inspection in SDNs," *Proc. Comput. Sci.*, vol. 131, pp. 1209–1216, Jan. 2018.
[71] S. Bae and S. Bae, "Big-O notation," in *JavaScript Data Structures and Algorithms: An Introduction to Understanding and Implementing Core Data Structure and Algorithm Fundamentals*. Apress, 2019, pp. 1–11.
[72] M. Buscema, "Back propagation neural networks," *Substance Use Misuse*, vol. 33, no. 2, pp. 233–270, Jan. 1998.
[73] X. Liang, Q. Tian, F. Wang, W. Yu, and X. Xin, "A dynamic resource allocation based on network traffic prediction for sliced passive optical network," in *Proc. 19th Int. Conf. Opt. Commun. Netw. (ICOCN)*, Aug. 2021, pp. 1–3.
[74] Y. He, Y. Yang, B. Zhao, Z. Gao, and L. Rui, "Network traffic prediction method based on multi-channel spatial–temporal graph convolutional networks," in *Proc. IEEE 14th Int. Conf. Adv. Infocomm Technol. (ICAIT)*, Jul. 2022, pp. 25–30.
[75] A. Cao, Y. Qiao, K. Sun, H. Zhang, and J. Yang, "Network traffic analysis and prediction of hotspot in cellular network," in *Proc. Int. Conf. Netw. Infrastruct. Digit. Content (IC-NIDC)*, Aug. 2018, pp. 452–456.

WAQAR ALI AZIZ received the B.Eng. degree in electrical engineering from the University of Engineering and Technology Peshawar, Peshawar, Pakistan, in 2016, and the M.S. degree in electrical engineering from the National University of Sciences and Technology, Pakistan, in 2020. He is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Cyprus. His research interests include the application of optimization methods in wireless networks, the use of metasurfaces in the next generation mobile networks, and deep learning in computer networks.

IACOVOS I. IOANNOU (Member, IEEE) received the Associate degree in computer science from the Cyprus College, in 2001, and the B.Sc. degree in computer science, the M.Sc. degree (Hons.) in computer and network security, and the Ph.D. degree with a focus on telecommunications in AI/ML from the University of Cyprus, in 2006, 2017, and 2021, respectively.

He is currently a Researcher with the Networks Research Laboratory, University of Cyprus. He is also a Junior Researcher with the Smart Networked Systems Research Group, RISE Center. His research interests include mobile and wireless communications, next-generation networks (5G), and device-to-device (D2D) communications, using artificial intelligence techniques. He is a highly skilled developer with 20 years of hands-on working experience in the IT industry, ranging the spectrum of IT systems from analysis, development, installation, and management. He was with the Philelefthеros Publishing Group as an IT Administrator and a Programmer for seven years, he was with Cyprus Stock Exchange as an IT and a Programmer for seven years and he was with Primetel for six years as an IT and Software Engineer with the Services Department. He has vast experience with cellular network infrastructure and all modern development platforms and languages. He has a certificate in ICONIX/SCRUM and he is also CCNET certified from CISCO.

MARIOS LESTAS received the B.A. and M.Eng. degrees in electrical and information engineering from the University of Cambridge, U.K., in 2000, and the Ph.D. degree in electrical engineering from the University of Southern California, in 2006. He is currently an Associate Professor with Frederick University. He has participated in a number of projects funded by the Research Promotion Foundation and the EU. His research interests include the application of non-linear control theory and optimization methods in complex networks, such as computer networks, transportation networks, power networks, molecular nanonetworks, and metasurfaces. In the aforementioned networks, he has investigated issues pertinent to congestion control, information dissemination, network vulnerability, demand response, and more recently privacy and security.

HASSAAN KHALIQ QURESHI received the M.Sc. degree (Hons.) in electrical engineering from the Blekinge Institute of Technology, Sweden, in 2006, and the Ph.D. degree in electrical engineering from City University, London, U.K., in 2011. He was a recipient of the EU Erasmus Mundus Staff Research Mobility and Postdoctoral Fellowship under the STRONG TIES and INTACT programs, respectively. He is currently a Professor with the School of Electrical Engineering and Computer Science, National University of Science and Technology (NUST). His research interests include wireless networks, the IoTs, cyber-physical systems, and blockchains.

ADNAN IQBAL received the Ph.D. degree in information technology from the National University of Science and Technology (NUST), in 2012. He is currently an Associate Professor with the Computer Science Department, Namal University, Mianwali. His research interests include software defined networks, wireless sensor networks, and cloud computing.

VASOS VASSILIOU (Senior Member, IEEE) is currently an Associate Professor with the Department of Computer Science, University of Cyprus, and the Co-Director of the Networks Research Laboratory (NetRL), UCY (founded by Prof. A. Pitsillides). Since November 2017, he has been the Group Leader with the Smart Networked Systems Research Group, RISE Center of Excellence on Interactive Media, Smart Systems and Emerging Technologies, Nicosia, Cyprus. He has also been appointed as the Senate of the University of Cyprus to the Board of Directors of CYNET, the National Research and Educational Network, where he serves as the Chair, since November 2016. His research interests include the protocol design and performance aspects of networks (fixed, mobile, and wireless), in particular mobility management, QoS adaptation and control, resource allocation techniques, wireless sensor networks, and the Internet of Things. He is a Senior Member of the ACM, the Treasurer of the IEEE Cyprus Section, the Chair of the IEEE ComSoc Cyprus Chapter, part of the IBM Academic Initiative, and an Academic Advocate for the ISACA Cyprus Chapter. He is also on the editorial boards of the *Telecommunication Systems* journal (Springer) and the *Computer Networks* journal (Elsevier), among others. He has been a member of the TPC for about 100 international conferences in his field and acted as the chair of various committees in more than 15 conference organizations.