Received 4 October 2024, accepted 18 November 2024, date of publication 25 November 2024, date of current version 4 December 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3506036

RESEARCH ARTICLE

SDN-Based Multipath Data Offloading Scheme Using Link Quality Prediction for LTE and WiFi Networks

SANTHOSHA KAMATH[1], J. ARAVINDA RAMAN[2], PANKAJ KUMAR[1], (Member, IEEE), SANJAY SINGH[1], (Senior Member, IEEE), AND M. SATHISH KUMAR[3], (Member, IEEE)
[1]Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, India
[2]Department of Computer Science, North Carolina State University, Raleigh, NC 27695, USA
[3]Department of Electronics and Communication Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104, India

Corresponding author: Sanjay Singh (sanjay.singh@manipal.edu)

ABSTRACT The continuous growth of mobile traffic and limited spectrum resources limits the capacity and data rate. Heterogeneous Networks (HetNet) is a solution with multiple radio interfaces in smartphones to realize such demands. Simultaneous data transfer on Long Term Evolution (LTE) and WiFi has gained attention for data offloading in 5G HetNet. Maintaining the average throughput and minimum delay for LTE users is still a challenge in data offloading owing to the mobility and load in the network. This study explores the benefits of Software-Defined Networking (SDN) based multipath for data offloading schemes for LTE-WiFi integrated networks to maintain the user's average throughput based on channel quality classification. We classify future link qualities using deep learning models such as Long Short-Term Memory Networks (LSTM) and Bidirectional Long Short-Term Memory Networks (BLSTM). The received signal strength indicator (RSSI) and packet data rate (PDR) are parameters used in BLSTM. The results of the prediction were compared with those of state-of-the-art methods. We obtained a 2.1% better prediction than the state-of-the-art methods. The predicted results were used to offload the data using LTE and WiFi. The performance of HetNet was compared with the state-of-the-art method for average throughput, and with the proposed method, a 6.29% improvement was observed.

INDEX TERMS Software-defined network, HetNet, mininet, floodlight, deep learning, LSTM, BLSTM.

I. INTRODUCTION

A cellular network is used everywhere, supporting various data-rich services such as multimedia applications, and video streaming, demanding more data rates. Due to the limited spectrum, it is difficult for service providers to manage such QoS and huge traffic demand. Investing in spectrum acquisition and buying devices is one of the solutions to this problem. Nevertheless, according to [1], it is expected that service providers will soon face the challenge of low revenue growth. Therefore rather than investing in the spectrum, most service providers are working intelligently to remunerate the traffic demand. Many techniques have been proposed by service providers and User Equipment (UE) manufacturers to fulfill these requirements.

Coordinated Multi-Point transmission (CoMP) [2] has been proposed to improve the reception performance of a UE at the edge of the cell. The surrounding cells cooperate with a specific UE to improve the performance by reducing interference. However, coordinating with neighboring enodB is only sometimes possible. To support CoMP, machine learning-based schemes such as Long Short-Term Memory Networks (LSTM) [3], which learns the characteristics of the network to predict coordinate information that maintains the average throughput of the UE, have been proposed. A behavior-aware beam forming technique [4] was proposed to increase data throughput in cellular networks. MIMO uses beamforming technology wherein multiple data streams are

The associate editor coordinating the review of this manuscript and approving it for publication was Tarcisio Ferreira Maciel.

transmitted through multiple antennas under the same carrier. The coexistence of radar and communication systems, which operate in overlapping frequency bands involves cooperative spectrum management. These strategies allow both systems to share the same resources efficiently without compromising their performance [5]. Despite these, challenges remain, particularly in scenarios with high traffic demands, such as video gaming or ultra-high-definition video streaming (e.g., 4K/8K). In these cases, the current technologies may not fully meet the user requirements for seamless and high-quality service, indicating the need for further research and innovation in the field.

Data offloading is one of the solutions to overcome the spectrum shortage problem [6]. Offloading refers to the use of complementary networks to deliver data initially targeted to cellular networks [7]. Most modern smartphones are equipped with Heterogeneous Networks (HetNet) interfaces; therefore, HetNet is cost-effective for offloading data.

Many HetNet concepts have been explored using Industrial, Scientific, and Medical (ISM) bands with LTE [8] for capacity enhancement. However, these techniques only use WiFi to offload data whenever both options are available. Similarly, it is observed that the average throughput of the user associated with WiFi reduces with the load in the network after a certain level [9]. Therefore, offloading partial data through both LTE and WiFi would be a better solution for maintaining the average throughput.

Network link quality also plays a vital role in QoS and decision on data offloading. Using low-quality connections would result in multiple re-transmissions. It can contribute to delivery delays and may also be unable to hide the unreliability of wireless connections, resulting in message losses. Consequently, the quality of experience (QoE) is affected, and the user can face some issues. Proactive optimization of wireless communication systems can be enabled by prior knowledge of channel quality with high precision and low overhead. For example, Channel Quality Prediction (CQP) has been proposed for optimal resource allocation. The objective is to classify the link quality using deep learning models with link quality metrics to precisely determine data offloading.

There are two types of metrics to estimate link quality in networks: hardware and software-based metrics [10]. One hardware-based metric is the Received Signal Strength Indicator (RSSI), which estimates the received signal power in a channel. The RSSI range describes the relationship between the transmitted power, received power of wireless signals, and distance between nodes. The register records the background noise in dBm when there is no transmission. The second parameter in the hardware-based metric is the Signal to Noise Ratio (SNR). It is a signal quality/strength measure or the difference in decibels (dB) between the received signal and background noise. It is used to compare the desired signal to the background noise level. The third parameter is the Link Quality Indicator (LQI). This metric indicates how well the data packets received by the receiver are. This statistic is used to determine the routing metric at the network layer.

The parameters used in software-based metrics are the Packet Delivery Ratio (PDR), Requested Number of Packets (RNP), and score-based parameters. The PDR is the ratio of packets sent by the source to the number of packets received by the destination. This is also called the Packet Reception Ratio (PRR). The RNP is a metric that calculates the number of transmissions/re-transmissions before successful packet reception. Score-based metrics assign a score or label to a link's quality without referring to a physical phenomenon. Some examples include the Fuzzy Link Quality Estimator, and Channel State Information (CSI).

The accuracy provided by hardware-based metrics is insufficient due to two primary issues. First, packets that are successfully transmitted are evaluated, and second, the assessment does not include the entire received packet, but simply its first symbols. RSSI is a hardware-based metric providing a quick and accurate estimate of whether a link is good quality or not [9]. However, it is inappropriate for use as a stand-alone metric for quantifying link quality because it does not capture the amount of destructive interference on the links. Due to the challenges associated with MAC coordination [11], machine learning approaches have been employed to predict channel quality. It is challenging, if not impossible, to accurately compute SNR in practice. We focus on the RSSI as a channel quality indicator and hardware component. The measurements and calculations involved in RSSI are less complicated, and the RSSI values are readily available from the chipsets. Therefore, we consider one hardware and one software metric for predicting the channel quality.

The network dataset comprises time-series data. Time-series data are a collection of observations obtained through repeated measurements over time. It comprises a multivariate time series augmented with a network structure. It describes the evaluation of a set of observations at the network nodes over time. Machine Learning (ML) and Deep Learning (DL) help in understanding and predicting link quality. LSTM [12] is mainly used for the classification and prediction of channel quality. LSTMs promise to learn the context required to make predictions in time series forecasting problems; they often do not require RNNs because a few recent events within a few small time windows convey all relevant information about the next event. Bidirectional Long Short-Term Memory Networks (BLSTM) are used for predictions by using information from the past and future via forward and backward sequencing. In this situation, the current information depends on past information and is linked to future data.

The main contributions of this paper are

1) Propose an LSTM and BLSTM-based channel quality prediction architecture using single and multiple parameters.
2) Flowlet-based multipath data offloading scheme for LTE and WiFi networks using link quality.

We implemented channel prediction and data offloading using a Software-defined Network (SDN), an emerging technology that separates data from the control plane. The data plane processes packet forwarding at the switch, and an SDN controller performs network control with a total view of the network using the most popular protocol, OpenFlow. OpenFlow defines the rules for communication between the controller and the switches. With SDN, the network operator can apply routing protocols and applications to the network quickly and flexibly. Utilizing the aforementioned advantages of SDN technology, we implemented an LTE-WiFi offloading system. This provides a high efficiency in data control traffic.

The remainder of this paper is organized as follows. Section II discusses related progress in the proposed research area. Section III provides details of about the proposed method. Section IV provides details of the experimental setup. Section V provides the experimental results. Section VI discusses the results and Section VII concludes the paper.

II. RELATED WORK

We divide the related work into channel quality prediction and data offloading.

A. CHANNEL QUALITY PREDICTION

Several CQP approaches have been extensively researched in various network contexts to allow for future channel improvements or prevent adverse conditions from affecting wireless communication systems. Liu and Cerpa [13] provided one of the earliest models for the Link Quality Estimator (LQE), in which they proposed three machine learning algorithms: naive Bayes, neural networks, and logistic regression, each of which provides a multi-class output. However, time information was not incorporated to predict link quality; therefore, dynamic link quality prediction was impossible.

Sindjoung and Minet [14] proposed the prediction of link quality using two metrics, RSSI and PDR, using ML algorithms such as logistic regression, linear support vector machine (SVM), and Random Forest classifier. One of the first deep learning approaches was to estimate link quality. The limitation was that the overall accuracy was low.

One of the most recent link quality classification models by Boucetta et al. [15] compared the KNN and LSTM-based models to classify the link quality into five classes using RSSI and PDR metrics. Diouf et al. [16] proposed channel prediction using LSTM and RNN-the validity of the proposed deep learning approach based on the root mean square error (RMSE). The performances in terms of RMSE with the same dataset for each of the models used in this study were compared to those of other models. It is observed that LSTM provides a low RMSE.

Schuster and Paliwal [17] presented a bidirectional LSTM (BLSTM) as an extension of the conventional LSTM. Nsaif et al. [18] used BLSTM for the Link-State Prediction for Software-Defined DCN Power Optimization. The goal is to reduce link utilization. However, the link state is predicted based on bandwidth utilization in the network.

Depending on the type, existing metrics often assure either stability or accuracy, but seldom both. Previous research has attempted to classify links and address link asymmetry. The intermediate connections are highly unreliable. In this context, we use hardware and software-based indicators, that is RSSI and PDR, to classify connection quality into good, intermediate, and bad classes, similar to some of the previous works [14], [15], but with improved prediction accuracy and performance. We used deep learning algorithms to achieve the same.

B. DATA OFFLOADING

The integration of LTE-WiFi can be classified into two types: network selection and network aggregation. The first approach can be performed without upgrading the LTE and WiFi networks. The complexity is the selection of the best network for data offloading. The second approach uses both networks to increase the data offloading capacity. Therefore, more sophisticated changes are required for both the LTE and WiFi.

Wang et al. [19] proposed an Intelligent Data Uploading Selection Mechanism for cellular networks. The authors reported that data offloading reduces cellular usage by 50% through the experiments. However, this work was performed without SDN, and the load on WiFi was not considered for offloading.

Network selection provides interfaces and intelligence to switch data between the two networks. Lee et al. [20] studied the economic aspects of data offloading using WiFi. They used a game-theoretic approach for data offloading. This scheme waits until the network is congested before the offloading is initiated.

Saliba et al. [21] explore strategies for leveraging WiFi to offload traffic from LTE networks within 5G environments, with the goal of enhancing overall network performance and user experience. The study emphasizes dimensioning techniques that account for user density and data demand to optimize WiFi deployment for efficient traffic management. However, it is important to note that this work does not incorporate SDN.

Deng et al. [22] proposed the *Delphi* method, which is a data offloading method based on best network selection. Delphi is a network selection protocol for the transport layer. This protocol selects the best network to fulfill user objectives. Anbalgan et al. [23] proposed a novel data offloading scheme called the SDN Assisted Learning Approach (SALA) for data offloading. They used multiple SDN controllers, one at LTE and another at WiFi. Controller-to-controller communication was proposed in this approach. However, it uses a single network at a time to offload data.

Ford et al. [24] proposed an MTCP, that uses multiple TCP flows into the transport layer. This method controls sub-flows and congestion in the network. Handover in a mobile scenario for an MTCP was proposed in [25]. However, all these proposals depend on the scheduler, which has a local network view. A global view of the network avoids greedy traffic scheduling and reduces the delays. Therefore, MTCP requires a control plane with a global view. Moreover, these ideas were built without SDN in their networks.

To optimize resource utilization and distribute processing across CPU cores using SDN, Biersack et al. [26] demonstrated how traffic management can be combined with load- and traffic-aware power management to reduce power consumption without compromising the ability to satisfy application requirements. It has an edge server connected to a network card, that maintains a holistic view and runs an algorithm to adapt to changes in network traffic.

An SDN-based control plane for the MTCP was proposed in the SD-MTOP [27]. This solves mobility issues between the two technologies. Adding control at the control plane helps the SD-MTOP configure the network flow. It uses a traffic steering method to offload data. The offloading solution determines the network that is connected to a given set of users. in this study, a throughput maximization problem was addressed using multipath. However, MPTCP cannot adaptively control subflows. The disorder of data packages caused by differences between paths leads to poor performance compared with single-path transmission. It assumes that a WiFi network is always available and 50% of the data are offloaded through WiFi. The load on WiFi affects the performance of the network [28], which is not addressed in this paper.

Chen et al. [29] proposed a new MPTCP subflow control algorithm (MSCA) based on a SVM prediction model. The SDN controller is used to continuously monitor the network status and predict the impact factor of the network based on the monitored path parameters. Subsequently, according to the impact factor, the subflow configurations are dynamically adjusted by the system for each user to improve the average throughput.

Most proposed methods for data offloading using LTE and WiFi use a network selection scheme. This means that they try to offload the data using WiFi whenever the WiFi range is available, because of the high data rate available at WiFi. However, it is observed that the average throughput of the user associated with WiFi decreases with the load in the network after a certain level [9]. Therefore, the WiFi load must be tested before data offloading or offloading through both the networks. Only a few works such as [27], use the network aggregation method in which the data are offloaded using both LTE and WiFi. However, this approach requires a multipath data transfer. Multipath data transfer such as Equal Cost Multipath (ECMP) [30], is well studied at data centers. The distance between different paths is smaller in the data center; therefore, packet reassembly is not a problem. However, for use cases such as LTE and WiFi, the advantage of multipath transmission is nullified owing to packet reassembly issues at the receiver. There is a need for a multipath scheme for data offloading using LTE and WiFi to address packet reassembly delays.

Similarly, due to the mobility of the users, there is a variation in the channel quality, which affects the user throughput. To maintain the user throughout, it is necessary to offload the data through WiFi and LTE whenever the channel condition is poor. Many studies have been conducted on the prediction of channel quality. However, channel quality-based multipath data offloading is yet to be explored.

III. PROPOSED METHOD

The proposed method consisted of two parts. First, machine learning predicts channel quality for a given application. Second, based on the predicted channel quality, a network architecture was provided to offload the data using LTE and WiFi. Table 1 lists the acronyms and notations used in this study.

TABLE 1. Acronyms and notations used.

LSTM	Long Short Term Memory
BLSTM	Bidirectional LSTM
PDR	Packet Data Ratio
PRR	Packet Reception Ratio
PRB	Physical Resource Block
RSSI	Received Signal strength Indicator
RSRP	Reference Signal Received Power
SD-MTOP	Software Defined- Multi Path Offloading Module
DCF	Distribute Coordinated Function
SLA	Service Level Agreement
w	Weight of neural Network
y_i	True label
$\hat{y}_i$	Predicted Label
K	Number of Simultaneous Users
L_i	Link throughput for ith user
$D_{\text{avg}}^{\text{WiFi}}$	Average throughput on WiFi
$D_{\text{avg}}^{\text{LTE}}$	Average throughput on LTE
V	Data Volume in-terms of Packets
T_{LTE}	Minimum time required to send the data via LTE
T_c	Threshold time for given application packets
V_{WiFi}	Data Packets through WiFi
V_{LTE}	Data Packets through LTE
N_{PRB}	Number of PRB
R	Transmission rate of a user in LTE with number of PRB

A. CHANNEL QUALITY PREDICTION

We used LSTM and BLSTM to classify and predict the channel quality. We investigated the parameters affecting the prediction of channel quality and explored the results obtained using LSTM and BLSTM.

1) DATA SET

We used the dataset available in the IoT-LAB [14] for the study. These data were collected from 50 nodes in 48 hours and linked over 16 channels. The sample dataset is as given in Table 2. First, we perform data pre-processing, which includes eliminating redundancy by downsizing and removing outliers. In addition, the missing values were filled with the mean values. Next, we calculated the standard deviation, mean, minimum, and maximum of each parameters, as shown in Table 3. As shown in Table 3, the standard deviation (std) and mean do not vary for the last parameter, Tx_count (the number of packets sent). So, we drop it.

TABLE 2. A sample dataset.

Date Time	Source	Destination	Chanel	Mean RSSI	PDR	Tx_count
2018-01-11 16:32:22	0	7	11	-70.74	1	100
2018-01-11 16:32:22	0	42	11	-80.06	0.6	100
2018-01-11 16.:32:2	0	28	11	-68.46	1	100

TABLE 3. Description of dataset.

	Source	Destination	Channel	Mean_RSSI	PDR	Tx_count
Count	108659.0	108659.0	108659.0	108659.0	108659.0	108659.0
mean	24.92	25.04	18.37	-74.55	0.87	100
std	14.83	14.83	4.57	11.87	0.24	0.00
min	0.00	0.00	11	-91.0	0.01	100
25%	12	13	14	-87.77	0.91	100
50%	25	25	18	-76.57	1	100
75%	39	40	22	-66.97	1	100
max	49	49	26	-5.90	1.0	100

TABLE 4. RSSI-based classification threshold.

Good	Intermediate	Bad
RSSI $\geq$-85	-85 $>$ RSSI $>$ -87	-87 $\leq$ RSSI

TABLE 5. PDR-based classification threshold.

Good	Intermediate	Bad
PDR$\geq$0.75	0.75$>$ PDR$>$0.3	PDR $\leq$ 0.3

TABLE 6. RSSI and PDR-based threshold for classification.

Class Label	Threshold
Good	$PDR \geq 0.75$
Intermediate	$0.75 > PDR > 0.3$ AND $-75 \leq RSSI > -87$
Bad	$PDR \leq 0.3$ OR $0.75 > PDR \leq 0.3$ AND $RSSI \leq -87$

2) DATA CLASSIFICATION

Three classes were defined: good (G), intermediate (I), and bad (B). We compared the classification based on hardware, software, and the combined parameters. Hardware classification was performed using only the RSSI. Furthermore, PDR was used for the software. The classification thresholds for RSSI and PDR are given in Table 4 and 5, respectively, and were adopted from [14]. We also studied the effect of the combined components of RSSI and PDR hardware and software metrics to compare the model's prediction. In combination, we have given PDR the first preference here because it considers all the packets sent, unlike RSSI, which only considers the received packets, as listed in Table 6.

3) FEATURE ENGINEERING, LEARNING, AND PREDICTION

Because the dataset is a time-series data, we considered daytime as the index. The date were split the data into 80% for training and 20% for testing. Table 2 shows that the RSSI and PDR have different ranges. For our model to accurately predict without giving preference to one of the variables with a higher range value, feature scaling [14] is required. In this study, we used *Standard Scaler* from *scikit-learn* [31], which is a machine learning library in Python to scale our variables. The *Standard Scaler* obtains a standardized distribution with a zero mean. It standardizes features by removing the feature's mean value and dividing the result by the standard deviation of the feature. Therefore, if we have a parameter that varies significantly, we scale it in a certain range, for example, [−1 1]. This was applied to both the training and test sets. A sample of the scaled training set is presented in Table 7.

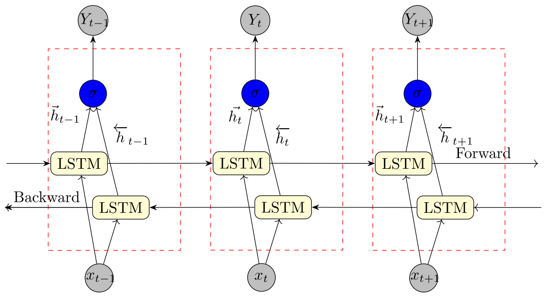


FIGURE 1. BLSTM-based RSSI and PDR quality prediction model.

TABLE 7. Cleaned dataset.

Date time	Mean_RSSI	PDR
2018-01-11 16:32:22	0.3212	0.5069
2018-01-11 16:32:22-	-0.4640	-1.0950
2018-01-11 16:32:22	0.5133	0.5059

After defining the classification parameters and, splitting the data into training and testing datasets, and scaled/cleaned datasets, we used bidirectional LSTM. We compared our results with those obtained using LSTM [15].

The Bidirectional LSTM (BLSTM) model used in this study has two parallel LSTM layers to produce a forward and backward loop, as shown in Fig.1. The BLSTMs connect two hidden layers to the same output layer. The forward layer output sequence, $\vec{h}$, is iteratively calculated using inputs in a positive sequence from time $T-n$ to time $T-1$. By contrast, the backward layer output sequence, $\overleftarrow{h}$, is calculated using the reversed inputs from time $T - n$ to $T - 1$. The BLSTM layer generates an output vector, Y_T, in which each element is calculated using the following equation:

$$Y_t = \sigma(\vec{h}_t, \overleftarrow{h}_t) \qquad (1)$$

where the σ function was used to combine the two output sequences. It can be a concatenating function, summation function, average function, or multiplication function. Similar to the LSTM layer, the final output of a BLSTM layer can be represented by a vector, $Y_t = [Y_{t-n}, \ldots, Y_{t-1}]$ in which the last element, Y_{t-1}, is the predicted network quality for the next iteration when considering network quality prediction.

For both models, we created sequences of T time steps, as this is a time-series dataset, and we need to predict the future quality. We considered a T value of 30; we looked back at 30 previous data values to predict the next value in the time sequence. Empirically, we observed that setting $T = 30$ was ideal for providing good accuracy for our data set. Setting a small value for T will restrict the LSTM to generalize the next prediction on a small set of values, while setting larger than

30 will provide more accuracy but consumes more time in training the model.

The architecture of the LSTM is defined with three hidden layers, eight neurons per layer, and an output layer with three neurons. Because there are three classes that we need to classify the link quality 13 epochs, as the loss converged after these many epochs. We also used *Batch Normalization* [32] after every LSTM layer to normalize the outputs of previous layers, reducing the number of epochs, making the learning more efficient, and avoiding over-fitting.

For bidirectional LSTM, we have two LSTMs in parallel; that is, two LSTMs are trained on the input sequence instead of one. Therefore, in this model, instead of the time-distributed layer receiving 30 timesteps (T = 30) of eight outputs, which we defined earlier, it will receive 30 timesteps of 16 (8 units + 8 units) outputs. Therefore, the three hidden layers will have 16 neurons, and the number of neurons in the output layer remains unchanged. To compile the model, we used categorical cross-entropy as a loss function, which is given by

$$J(w) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right] \quad (2)$$

where w is the weight of the neural network, y_i is the true label and $\hat{y}_i$ is the predicted label. Using these weights, we calculated the categorical loss using Equation 2 for each input feature. We updated the weights for each feature to minimize the error/loss for the predicted value using backpropagation and the Adam optimizer [32]. The predicted result was used to determine data offloading.

B. SDN ENABLED ARCHITECTURE FOR DATA OFFLOAD

We assume that the HetNet environment has a cellular network and WiFi, and there is mutual agreement between them or the WiFi network of the cellular network. In addition, we assume that the UE has multiple interfaces to support HetNet communication. We used two SDN controllers, one on the cellular side, which controls the operation of the cellular network, and another to control the WiFi operation as shown in the architecture (see Fig 2). This is because a single SDN controller has several disadvantages in terms of scalability and performance. However, the use of multiple controllers increases the complexity. We must establish communication between the controllers. We used Opendaylight controllers on both sides, which is an open-source platform that supports clustering. To offload the data, we need specific information, such as the capacity of the WiFi and the cellular network. This information is obtained using [33]. The controller application of the LTE will have an authentication and charging module for identity verification and to maintain the charging account. We included a Data Offloading (DO) module to track data rates in WiFi and cellular networks. There is an application identification module [34] that provides the characteristic features of the application, such as the IP addresses, QoS information bit rate, and port number used as per the traffic flow template (TFT) [35]. Once the authentication is successful, resources are allocated to the user.

The resource monitoring component continuously measures the LTE network's RSSI and PDR and predicts the class of the network as discussed in Section III-A. The measured result was used to determine the data offloading given in the next section.

TRAFFIC OFFLOADING

We aggregated the spectrum by sharing data between the WiFi and the cellular network. Our approach uses LTE as the primary option to offload data. If sufficient channel quality is unavailable or the network is congested for the application, we use WiFi and LTE for offloading using multipaths. This means that we offload the maximum amount of data using LTE and the remaining data using WiFi. It is observed that the average throughput of the user associated with WiFi decreases with an increase in the network load after a certain level [28]. Therefore, it is necessary to consider the load on the WiFi network before offloading the data to ensure that the average data remain within a specific level.

WiFi follows a Distributed Coordinated Function (DCF), which means that when all users have a similar amount of traffic, they have fair, medium access, that is, a max-min fairness system. Therefore, we have adequate capacity sharing for all users within a single WiFi network. Let L_i be the link throughput for the i^{th} WiFi user, and the average throughput D_{avg}^{WiFi} per user, assuming K simultaneous users are accessing the network, is given by

$$D_{\text{avg}}^{\text{WiFi}} = \frac{1}{K}\sum_{i=1}^{K} L_i. \quad (3)$$

Let R_{LTE} be the data rate for a given LTE user, which depends on the bandwidth B of a channel and is given by

$$R_{\text{LTE}} = B * \log(SNR + 1) \quad (4)$$

where SNR is the signal-to-noise ratio measured over the entire bandwidth B. In LTE, the capacity of the user depends on the number of physical resource blocks (PRB), N_{PRB}, and the channel bandwidth assigned to an individual user. The transmission rate of a new user is given by

$$R = R_{\text{LTE}} * N_{\text{PRB}} \quad (5)$$

Because the standard is not specified in LTE, and if we assume that LTE also provides fair access to every user, a user with worse channel conditions is provided with more PRB. The average throughput $D_{\text{avg}}^{\text{LTE}}$ available to the user is given by

$$D_{\text{avg}}^{\text{LTE}} = \frac{1}{K}\sum_{i=1}^{K} R_i \quad (6)$$

where R_i denotes the maximum link throughput for i^{th} user. Therefore, the total time required to transfer the V number of

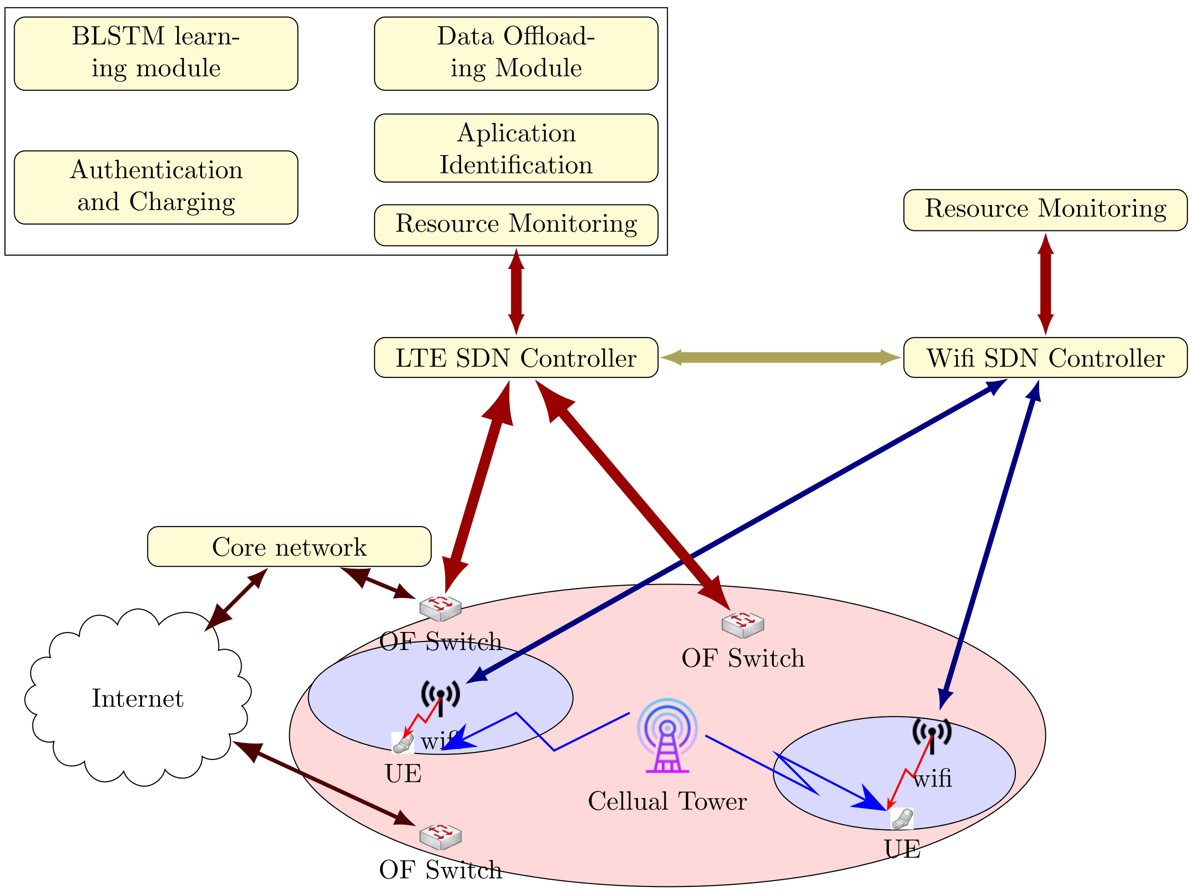


FIGURE 2. SDN-based heterogeneous network architecture.

data packets at LTE alone is given by

$$T_{\text{LTE}} = \frac{V}{D_{\text{avg}}^{\text{LTE}}} \tag{7}$$

Let us assume T_{WiFi} is the time required to transmit data in WiFi at a given throughput $D_{\text{avg}}^{\text{WiFi}}$ and T_c is the threshold for sending the data to the ith user. T_c is obtained based on the SLA or by predicting the application behavior using the application identification module (Fig 2). If $T_c > T_{\text{LTE}}$, then complete data cannot be transmitted over the LTE. The amount of data that LTE can transmit, assuming maximum utilization of LTE, is given by

$$V_{LTE} = D_{\text{avg}}^{\text{LTE}} * (T'_{\text{LTE}}) \tag{8}$$

where

$$T'_{\text{LTE}} = T_c - T_{\text{LTE}} \tag{9}$$

The remaining data needs to be offloaded through the WiFi, which is given by

$$V_{\text{WiFi}} = V - V_{\text{LTE}} \tag{10}$$

Based on $D_{\text{avg}}^{\text{LTE}}$, T_{LTE} and T_c our architecture decides on data offloading. If $T_c < T_{\text{LTE}}$, WiFi offloading will not occur. The data rate depends on the number of PRBs, the bandwidth assigned, and the RSSI. Due to the mobility of the nodes, the signal strength varies. Hence, RSRP varies, which is calculated as

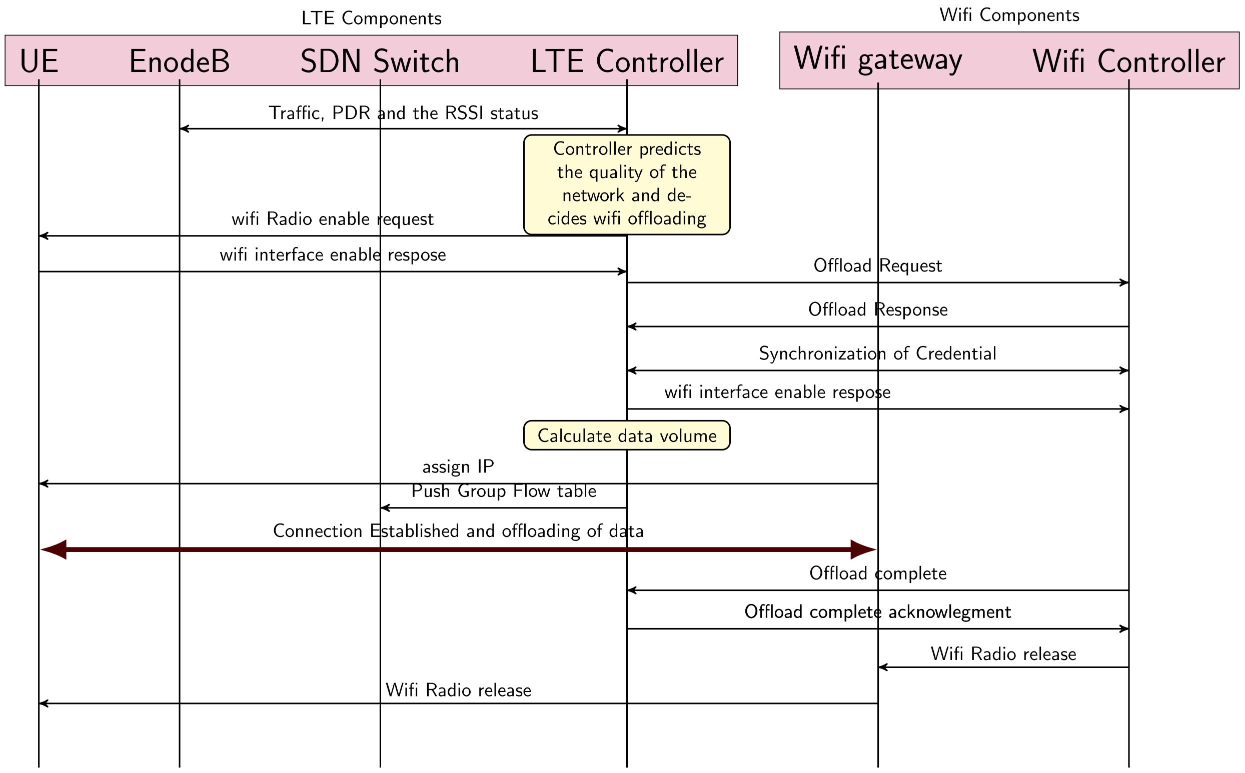
TABLE 8. RSSI and PDR-based threshold for video classification.

Class Label	Threshold
Good	$PDR \geq 0.75$
Intermediate	$0.75 > PDR > 0.3$ AND $-80 \leq RSRP > -90$
Bad	$PDR \leq 0.75$ OR $PDR \leq 0.3$ AND $RSRP \leq -90$

$$RSRP = RSSI - 10log(12 * N_{\text{PRB}}) \tag{11}$$

It is necessary to offload the data through WiFi to maintain the required data rate for a given application. The controller collects information about the RSSI and PDR of the cellular networks at regular intervals. If the value falls into the threshold class, WiFi offloading is initiated. We updated the classification parameters given in Table 6 for the video signal as per [36], and the updated values are listed in Table 8. Section VI provides a detailed discussion of the threshold selection.

Algorithm 1 is written to offload the data from LTE under two cases:



LTE Components
Wifi Components
UE
EnodeB
SDN Switch
LTE Controller
Wifi gateway
Wifi Controller
Traffic, PDR and the RSSI status
Controller predicts the quality of the network and decides wifi offloading
wifi Radio enable request
wifi interface enable respose
Offload Request
Offload Response
Synchronization of Credential
wifi interface enable respose
Calculate data volume
assign IP
Push Group Flow table
Connection Established and offloading of data
Offload complete
Offload complete acknowlegment
Wifi Radio release
Wifi Radio release

FIGURE 3. Data offloading message exchange between LTE and WiFi controllers.

i) If $T_{\text{LTE}} < T_c$, then it initiates the data offloading through both LTE and WiFi.
ii) If $T_{\text{LTE}} > T_c$, then offloading takes place through LTE only. It measures the RSSI and PDR and predicts the channel quality. If the predicted channel quality falls within the bad threshold as per Table 8, then in addition to LTE, it also initiates the data offloading via WiFi.

1) COMMUNICATION BETWEEN CONTROLLERS

The message exchange between the two controllers is illustrated in Fig 3. The LTE-SDN controller controls all components of the LTE and collects information such as traffic status, PDR, and RSSI. When Algorithm 1 initiates traffic offloading, the process is initiated by activating WiFi radio to the UE. The UE identifies the nearest WiFi access point and forwards this information to the LTE controller. Upon receiving this information, the LTE controller initiates a request to the WiFi controller and requests the current load in the WiFi. Subsequently both controllers exchanged the credentials of the UE. The LTE controller determines the amount of data offloading through the WiFi using Equation 10, as given in Algorithm 1, which requires data splitting. Traffic splitting for multipaths is presented in Section III-B3. The LTE controller pushes the group Table for data offloading, as listed in Table 9. The WiFi controller selects the WiFi gateway, allocates WiFi radio to the activated WiFi interface for the selected UE, and assigns an IP address to the UE WiFi interface. Once the connection is established, data offloading occurs through WiFi and the UE. Finally, upon completion, the WiFi controller releases resources and acknowledges the LTE controller.

2) FLOW DETECTION

The total file size can also be obtained during the HTTP handshake using a header called *Content-Length* from the HTTP response if the network has access to the L7 protocol. However, our study was limited up to the L6 protocol. We polled the flow entries in the edge switch to determine the sent byte counts. Each polling flow entry is uniquely defined, and consists of four tuple fields: source IP, destination IP, source transport port, and destination transport port. Using this, we calculate the rate at which it arrives by reading the count for every second at the edge switch and compare it with Equation (4). If it is above threshold (T_c), we fetch the HetNet offloading and assign it to V.

3) TRAFFIC SPLITTING

If $T_{\text{LTE}} > T_c$, then data must be offloaded through LTE and WiFi; for that, we must split the traffic between two paths at the edge switches. Randomly splitting the traffic

Algorithm 1 Data Offloading Algorithm

```
Input: Fetch RSSI, PDR, Bad-class Threshold, $V$, $T_c$

1:  while end of the data volume do
2:    Average WiFi load ← $D_{avg}^{WiFi}$ using (3)
3:    Calculate $T_{LTE}$ using (7)
4:    Calculate $V_{LTE}$, $V_{WiFi}$ using equation (10) and (8)
5:    if $T_{LTE} < T_c$ then
6:      Goto step 11
7:    end if
8:    Offload the data $V$ using LTE networks.
9:    Channel_quality= Read RSSI and PDR and predict the
      quality of the channel using BLSTM
10:   if Channel_quality = BAD class (Ref Table 8) for a
      given application then
11:     Enable Controller to controller communication
12:     Initiate UE to connect to WiFi and offload $V_{WiFi}$
        amount of data and update $V$
13:     Calculate flowlet using Algorithm 2
14:     Deploy the bucket action at the edge switch
15:   end if
16: end while
```

leads to a packet reordering problem at the receiver; thus the expected QoS cannot be achieved. We used flowlet [37] based routing at the edge switch, which can be measured using piggybacked packets. A flowlet is characterized by Δ, the minimum inter-flow spacing between the packets. The Δ value is $\Delta \geq |t_1 - t_2|$, where t_1 and t_2 are the delays on two different paths to reach the destination. Due to the flowlet concept, consecutive flowlets can be switched independently without the risk of packet reordering. The value of Δ can be assigned by taking the difference between the maximum and minimum path delays. Algorithm 2 was used to split the traffic in an SDN-based edge switch.

The traffic splitting algorithm applies to the packet that matches with the application identified by the Application Identification (AI) module (see Fig 2). First, it calculated the flowlets based on the Δ. It calculates the time gap between two consecutive packets (new_time to last_time). If the time gap is greater than Δ, Algorithm 2 considers it to be a new flowlet. Next, if there is a change in the flowlet, the algorithm calculates the forward path. In our case, we only have two paths: one through LTE and the other through WiFi. We generate a random number between 0 and 1 based on the data rate and find the flowlet's frequency (a_1 and a_2 are some intermediate variables in Algorithm 2). A packet is transmitted over the previous link if there is no change in the flowlet.

For example, the data rate must be split into two parts: LTE (with 3 Mbps) and WiFi (with 2 Mbps). We calculated the frequency as a_1 (0, 0.4] and a_2 (0.4, 1] and generated a random number between 0 and 1. If the number falls within the a_1 range, the flowlet is transmitted to the selected path based on the *flowlet* value. Over time, the packets were divided into a 3:2 ratio.

Algorithm 2 Traffic Splitting Algorithm

```
Input: Number of outgoing paths $n = 2$ and PathIDs
$\{p_1, p_2\}$ its capacities for the selected flow$\{D_{avg}^{LTE}, D_{avg}^{WiFi}\}$
respectively.
$D = D_{avg}^{LTE} + D_{avg}^{WiFi}$
last_time ← 0
packet_id ← hash(packet)
Output: PathId
1:  for Every packet enters into the ingress switch do
2:    if Packet_id matches then
3:      new_time ← get_time()
4:      if (new_time-last_time )> Δ then
5:        flowlet=1
6:      end if
7:      last_time = new_time
8:      calculate the frequency of the flowlet to be sent
          $a_1 = \left(0, \frac{D_{avg}^{LTE}}{D}\right], a_2 = \left(\frac{D_{avg}^{LTE}}{D}, \frac{D_{avg}^{WiFi} + D_{avg}^{LTE}}{D}\right]$
9:      if (flowlet==1) then
10:       flowlet=0
11:       $r = random()$
12:       if r ≤ $a_1$ then
13:         new_PathID= $p_1$
14:       end if
15:       if $a_1 \leq r \leq a_2$ then
16:         new_PathId=$p_2$
17:       end if
18:       forward packet to the link new_PathId
19:     end if
20:   end if
21: end for
```

The flowlet detection module is implemented in OpenFlow switches and the controller pushes the multipath decision as a group table [38]. Table 9 presents the bucket actions for the group table. The bucket action in the table is selected based on Algorithm 2. The *flow table* in OpenFlow is defined as a pipeline of stages. Each stage has a match and a corresponding action. It is forwarded to the controller if no instructions are available or if it reaches its destination. A group table can be used when there is a need for one of several sets of last minute actions to be applied to a packet or several copies of the packet to be sent, each with its last minute actions applied. For example, suppose it is necessary to perform multiple actions, such as making two copies of data and sending one to VLAN and another to the network analyzer. In this case, each packet requires a different encapsulation. To achieve this, it is necessary to modify the rules in all flow tables to create copies. Instead, create an *ALL group* to represent the destination and all of the

TABLE 9. Group entry for HetNet at LTE edge switch.

group_id	group_type :INDIRECT	bucket_action
group_id=1	select	bucket=1 action =OUTPORT:p_1 bucket=2, action=OUTPORT:p_2

relevant analysis, where each entry has the required actions to apply to that particular copy, such as adding a VLAN and setting the relevant VLAN-ID. Now, the flow tables are sent to this group, and if the analyzer is added or moved, it is not necessary to modify the flow tables; instead, modify the group.

IV. EXPERIMENTAL SETUP

We implemented our proposed data offloading method using a recent version of the Mininet-WiFi extension [27], which has a virtualized WiFi interface on wireless devices capable of emulating WiFi mobile stations. It also models the wireless physical channel characteristics and host movement. It also supports mobility models such as customized movement, random walking, and random waypoints. Our Open vSwitch (OVS) is installed to emulate SDN edge devices that communicate with the SDN controller using the OpenFlow protocol. The parameter configured for remote radio heads is given the Table 10. The experimental setup is shown in Fig.4. The SDN controller interacts with the mininet WiFi controller and collects information about the WiFi network. A link with capacity of 1 Gbps is used to connect internal components in WiFi and LTE respectively. Mobile hosts are equipped with WLAN and LTE interfaces to support the HetNet. Algorithm 1 was implemented using python in the data offloading module and BLSTM in the controller. The controller continuously monitors the network status and computes data offloading using Algorithm 1.

For the simulation, we used one WiFi and one LTE base station. Ten nodes, N_1 to N_{10} are deployed. Randomly, they connect to the LTE or WiFi networks. Node N_2 has dual interfaces, which means that it can connect to both WiFi and LTE. We assume that UE (N_2) is active, and only one 10 MHz bandwidth is available under ideal radio conditions. PRB (N_{PRB}) comprises 100 blocks, of which more than 80% of the resource blocks are available for node N_2. The other nodes have only one interface: either LTE or WiFi. We consider the application 1080p video, which requires 5 Mbps assuming an RTT of 50ms at an RSRP of -90dBm [39]. To test the effect of the load on the WiFi, nodes N_5 to N_9 communicate through WiFi.

`Iperf` command was used for the throughput tests, which measures the available bandwidth between two points of the network. We measured the average throughput of the network for video by generating an input traffic of 10 Mbps from the source (host) system to the destination N_2 node and the average output at different time intervals. Node N_2 is made to move away from position N_2* to N_2 as shown in Fig 4, from 0 to 50 seconds. Between 30 and 40 seconds, it reached

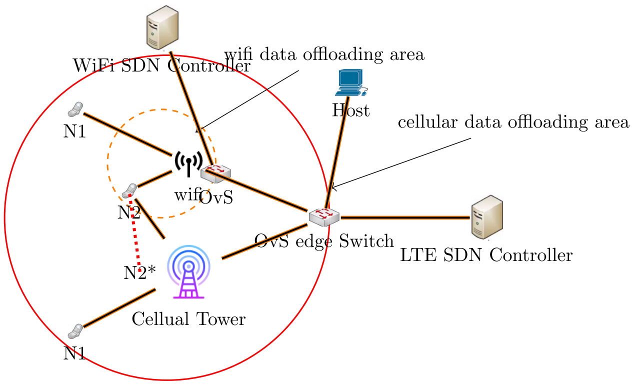


FIGURE 4. Experimental topology.

the WiFi range. Due to a change in the RSSI and PDR, the BLSTM module predicts and informs the data offloading module regarding the need for data offloading. The offloading data module at the LTE controller is then decided as per Algorithm 1. We evaluated the average throughput for our approach with SD-MTOP [27] in two cases. First, for a low load at WiFi and second, for a high load at WiFi.

TABLE 10. Simulation parameters.

Parameter	Values
Channel Bandwidth	20 MHz
Channel Type	AWGN, Flat Rayleigh
Modulation Scheme	16QAM
Frame period	10ms
User speed	10 KM/Hr
WiFi bandwidth	10 MHz
Number of PRB	100
Environment	Macro cell, Urban area
Cell range	500 m
PathLoss Model	$L = 128.1 + 37.6log(R)$
Carrier frequency	2 GHz
UE noise figure	9 dB
BS antenna Antenna height	30 m
Antenna maximum gain	15 dBi
Maximum BS power	46 dBm

V. EXPERIMENTAL RESULTS

The link prediction and data offloading performance results are given in this section.

A. RESULT FROM PREDICTION MODEL

We plotted the confusion matrix for both LSTM and BLSTM for single and double variables. To compare our results, we used one of the state-of-the-art methods given in [14] using LSTM, and our approach using BLSTM. Fig. 5 depicts the confusion matrix obtained with the traditional LSTM and BLSTM models applied to PDR, which is a software metric after labeling the classes based on Table 5. We obtained accuracies of 99.25% and 99.5% with LSTM and BLSTM, respectively. With BLSTM, only 0.15% of 'true bad' links are predicted as 'intermediate' compared to the traditional

LSTM, which is 0.26%. In addition, with LSTM, 0.3% of 'truly good' links were predicted as 'intermediate'; however, it was 0.0% with BLSTM. Moreover, both models predict a 4.2% error in intermediate class prediction.

Figure. 6 provides the confusion matrix using only the RSSI. Here, we can notice that 8.2% of the 'true bad' links were predicted as 'intermediate,' 0.0% of the proper intermediate links were predicted as bad, and 0.0% of the 'truly good' links were predicted as intermediate using LSTM. We noticed that the prediction error has slightly decreased in BLSTM. The 7.9% of 'true bad' links were classified as 'intermediate.' Only 0.12% of 'true intermediate' links were classified as 'good,' and 0.01% of 'truly good' links were classified as 'true intermediate.' So overall accuracy and performance of the BLSTM model for link quality prediction is better when compared to the traditional LSTM model for an RSSI component.

B. PREDICTION AND ACCURACY USING BOTH RSSI AND PDR

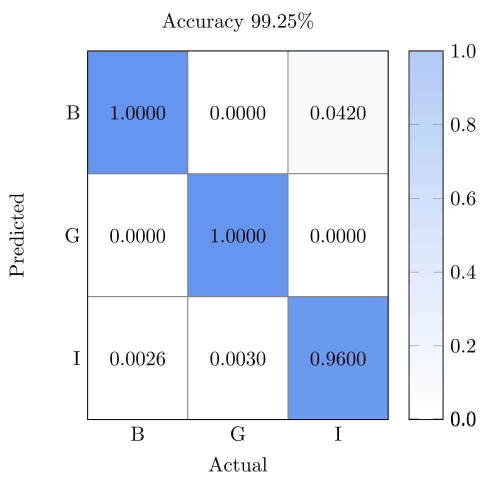
We used both RSSI and PDR to predict the link quality of the network to observe whether the combined effect yielded a better result than the metrics used individually. Figure 7a depicts the confusion matrix obtained with the traditional LSTM using the RSSI and PDR as features after labeling the links into three classes and classifying them. We obtained 99.73% accuracy. It is also noticeable from the confusion matrix that only 1.2% of actual wrong links were predicted as intermediate, which is better than the version using only RSSI for prediction, where 8.2% of actual bad links were predicted as intermediate. Additionally, only 1% of the intermediate links were classified as harmful.

Figure 7b depicts the confusion matrix obtained with BLSTM using RSSI and PDR as features after labeling the links into the three classes and classifying them. We obtained 99.94% accuracy. It is also noticeable from the confusion matrix that only 0.36% of the true bad links were classified as intermediate, and only 0.28% of the true intermediate links were classified as bad. Therefore, using the RSSI and PDR for link quality prediction is better than using only one feature.

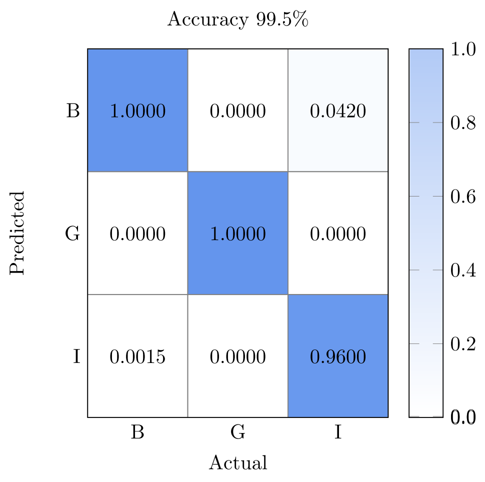
C. RESULT FROM DATA OFFLOADING

Intially, we tested the throughput performance of LTE only with our approach, as illustrated in Fig.8. The results clearly demonstrate that the multipath technique provides significant advantages for high data rate application transfers. Figure 9 shows a throughput comparison for a low load. When the node reaches the WiFi networks, our approach is capable of offloading the data to maintain an average throughput of 5 Mbps by offloading the data through WiFi and LTE together, which can be observed between 35 and 48 seconds. In contrast, SD-MTOP initiates offloading through WiFi at 50%. Similarly, the MTCP requires time to learn and use a single network. However, we could not find a significant difference in throughput between the MTCP and our approach at a low load. The only difference we can observe between 40-50 seconds is that SD-MTOP and MTCP take more time to decide the offloading and a small drop in average throughput near 40 seconds. Similarly, we can observe the variation when the output returns to normal quicker (between 40 and 50 Seconds) in our approach than MTCP. It is due to the prediction module we have used. For a better understanding of the throughput, we have plotted the cumulative distributed function (CDF) as shown in Fig. 10 shows that our approach is better due to the prediction, which will not allow dropping the throughput below the threshold.

We also evaluated the throughput at a high load using WiFi. We ensured that WiFi was loaded at more than 50% of the load by communicating with other nodes using WiFi. Furthermore, we repeated the same experiment for a low load, and the results are shown in Fig. 11. Because our approach considers the load on WiFi, it forwards data accordingly. However, SD-MTOP splits 50% of the data.

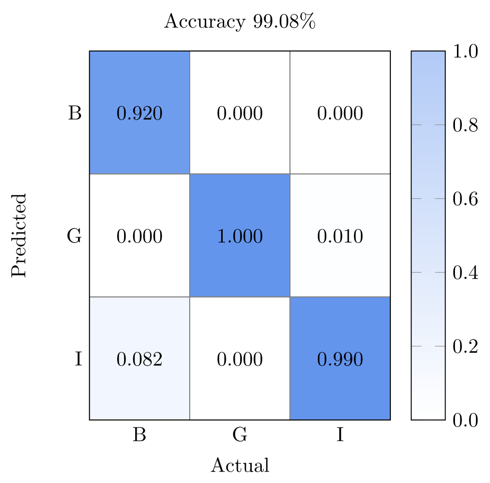


(a) Confusion matrix using LSTM

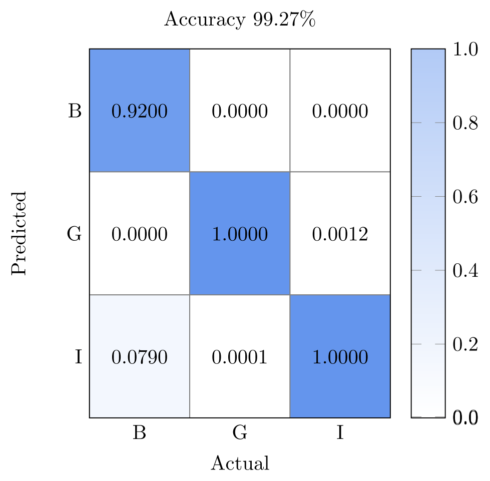


(b) Confusion matrix using BLSTM

FIGURE 5. Comparison of PDR-based confusion matrix.

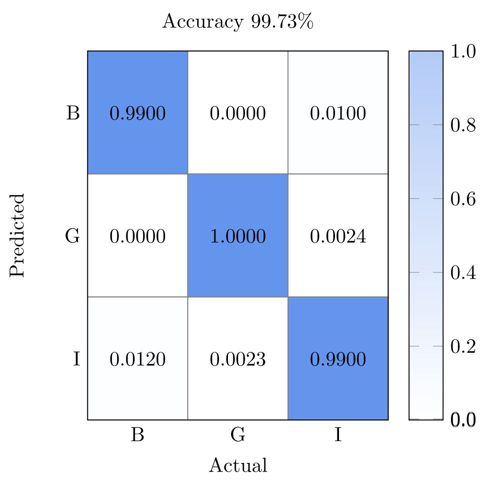


(a) Confusion matrix using LSTM

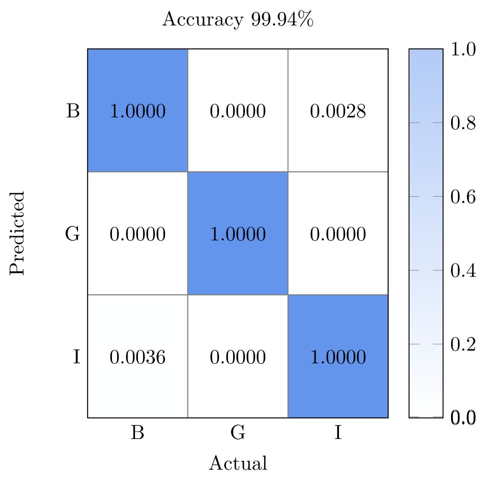


(b) Confusion matrix using BLSTM

FIGURE 6. **Comparison of RSSI-based confusion matrix.**



(a) Confusion matrix using LSTM



(b) Confusion matrix using BLSTM

FIGURE 7. **Comparison of confusion matrix for both parameters.**

Moreover, reordering TCP applications reduces the throughput in SD-MTOP, and our model solves this using a flowlet multipath. It can only guarantee the average throughput if WiFi is lightly loaded, and the RSSI is below the threshold. A throughput difference is observed in the CDF, shown in Fig. 12. In MTCP, the reordering issue were not considered. Therefore, our approach shows much better performance than other state-of-the-art methods.

VI. DISCUSSION

Prediction using BLSTM results is much better than LSTM for single parameters, as well as for combined parameters. From Fig. 5 and Fig. 6, it can be seen that overall, BLSTM is 0.05% and 0.19% better for PDR and RSSI, respectively, and from Fig 7, it is 0.21% for both the parameters in prediction than LSTM.

Comparing Fig. 5 and Fig. 6, we can conclude that PDR is a better metric to consider when compared to RSSI for link quality prediction. The number of ‘true bad’ links classified as ‘intermediate’ is only 0.26% when using only PDR, and 8.2% when using only RSSI. There can be a negative impact on the quality of experience (QoE) if more bad links are classified as intermediate. We can also say that PDR-based forecasts are more trustworthy than are RSSI-based forecasts, because PDR evaluates all packets delivered across the connection, whereas RSSI only examines successfully received packets. It would be interesting to study forecasts based on both the PDR and RSSI. By doing so, we will also be able to address one of the issues raised by Baccour in [9], who said that *“forecasting link quality requires finding an ideal trade-off between the estimator’s stability and its capacity to cope with link quality dynamics.”*

We also estimated the time required to train these models. The LSTM model took 1286.78 seconds without any GPU to train using both parameters. We obtained an accuracy of 99.73%, which is better than that obtained using RSSI and

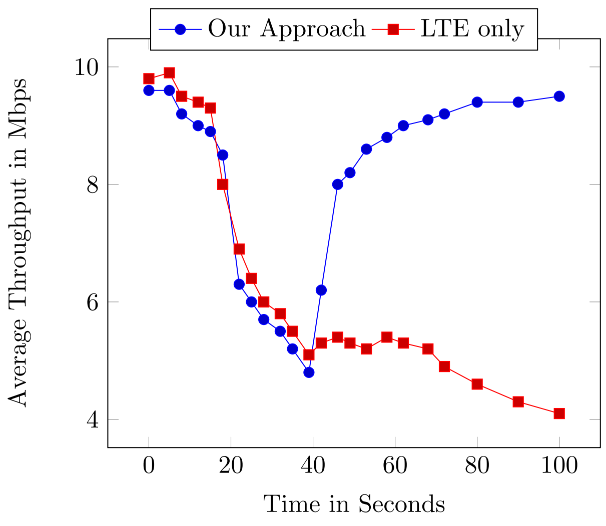


FIGURE 8. Throughput comparison of LTE only with our approach).

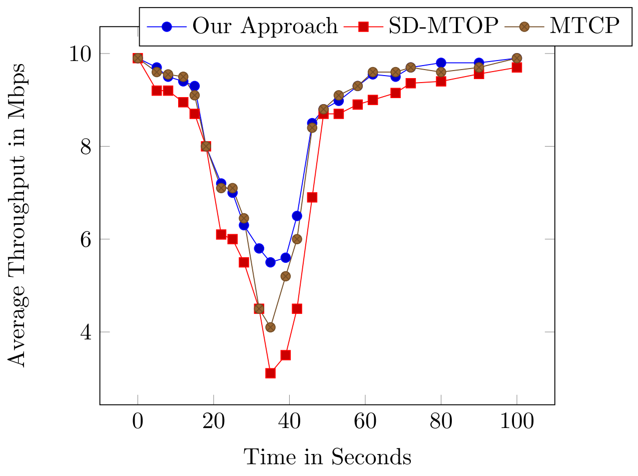


FIGURE 9. Throughput comparison of SD-MTOP, MTCP with the proposed method at low load (with minimum threshold throughput of 5 Mbps).

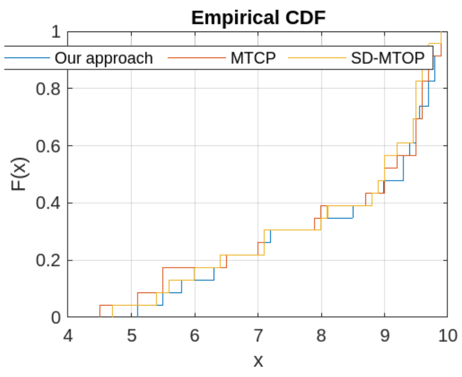


FIGURE 10. CDF comparison of SD-MTOP, MTCP with the proposed method at low load.

PDR individually for classification. On the other hand, the BLSTM model took 1618.70 Seconds to train without any GPU, and we obtained an accuracy of 99.94% on the test set, which is better than the traditional LSTM model's prediction.

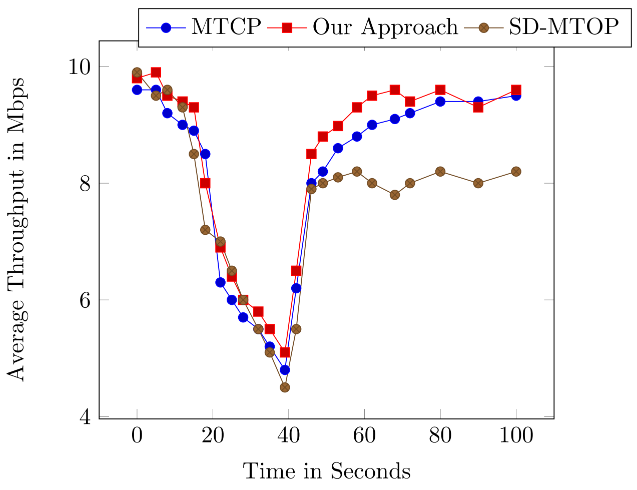


FIGURE 11. Throughput comparison of SD-MTOP, MTCP with the proposed method at high load (with minimum threshold throughput of 5 Mbps).

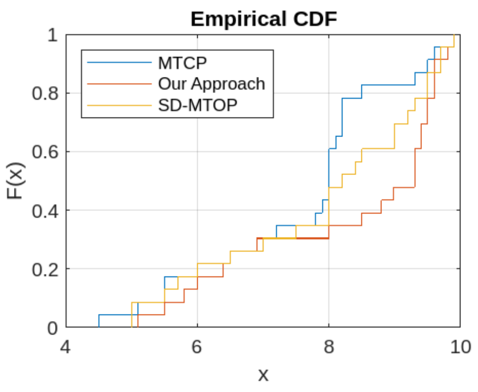
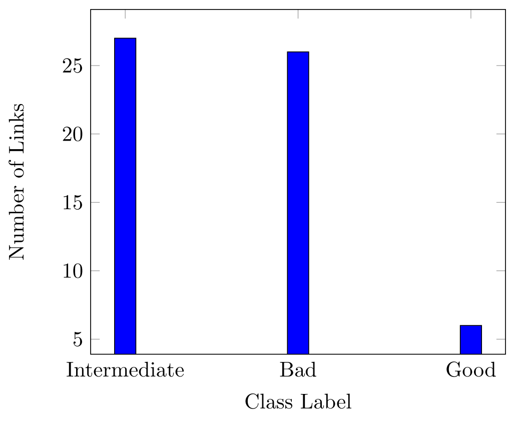


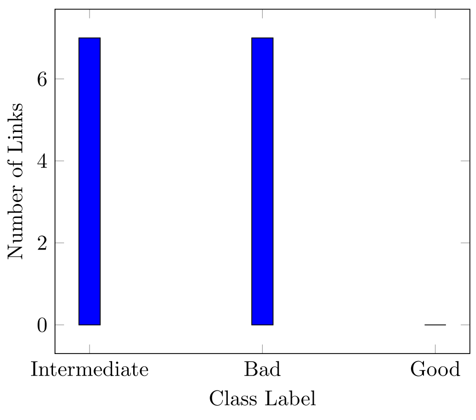
FIGURE 12. CDF comparison of SD-MTOP, MTCP with the proposed method at high load.

Figure 13 shows the number of misclassified links with their channels for the LSTM [14] and BLSTM. A total of (including I, B, G) 59 out of 21734 links were classified incorrectly in LSTM, and from Fig. 13b, it is just 14 for BLSTM. Most of the misclassified links belonged to either the bad or intermediate classes. Most of the links in our training set belong to the 'good' quality class, and there are very few 'bad' and 'intermediate' quality class links for our model to learn from and classify future data. Accuracy can be improved by choosing a dataset with an equal number of links in each class.

The observation shown in Fig.13 can be used as a threshold for data offloading. If the link is 'good,' the chances of predicting it as 'bad' or 'intermediate' are almost zero. Therefore, we can use the 'bad' class as the threshold. That is, if the PDR is between 0.75 and 0.3, and the RSRP is less than −90 dB, means the network is either 30% or 75% congested, with an RSSI equal to −87dBm, which is not suitable for transferring high data rate applications such as video [36]. Even though PDR does not directly affect the user but affects a provider, a low PDR indicates to the service provider that the network is congested to make the offloading decision.



(a) For LSTM



(b) For BLSTM

FIGURE 13. Comparison of miss classified labels.

With the proposed approach, the average throughput result obtained at low load is 1.73% (see Fig 9), and under high load, it is 6.29% (see Fig 11) better than that of SD-MTOP for the input of 10 Mbps.

VII. CONCLUSION

The rapid growth of data in mobile networks has lead to a massive overloading of traffic in cellular networks. Network congestion and channel quality are the essential components of data transmission. This paper presents BLSTM-based channel quality classification and prediction using software and hardware-based parameters such as PDR and RSSI. Using the predicted link quality, we provided a method to offload LTE data for high data rate applications, such as video using SDN, when channel quality and network congestion fulfill a user's QoS requirement. A flowlet-based multipath algorithm is presented, which considers the quality of the channel and the congestion in the network and offloads the data when the data are vast, which cannot be sent only via LTE because of congestion and poor network quality. The proposed method performed better than state-of-the-art methods.

REFERENCES

[1] (2023). *The Mobile Economy 2023*. [Online]. Available: https://data.gsmaintelligence.com/api-web/v2/research-file-download

[2] (2023). *4G LTE CoMP, Coordinated Multipoint*. [Online]. Available: https://www.electronics-notes.com/

[3] Y. Zhou, Z. M. Fadlullah, B. Mao, and N. Kato, "A deep-learning-based radio resource assignment technique for 5G ultra dense networks," *IEEE Netw.*, vol. 32, no. 6, pp. 28–34, Nov. 2018.

[4] D. Cong, S. Guo, S. Dang, and H. Zhang, "Vehicular behavior-aware beamforming design for integrated sensing and communication systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 6, pp. 5923–5935, Mar. 2023.

[5] C. Liu, Y. Chen, and S.-H. Yang, "Deep learning based detection for communications systems with radar interference," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6245–6254, Jun. 2022.

[6] Q. Chen, G. Yu, A. Maaref, G. Y. Li, and A. Huang, "Rethinking mobile data offloading for LTE in unlicensed spectrum," *IEEE Trans. Wireless Commun.*, vol. 15, no. 7, pp. 4987–5000, Jul. 2016.

[7] A. Aijaz, H. Aghvami, and M. Amani, "A survey on mobile data offloading: Technical and business perspectives," *IEEE Wireless Commun.*, vol. 20, no. 2, pp. 104–112, Apr. 2013.

[8] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: How much can WiFi deliver?" in *Proc. 6th Int. Conf.*, New York, NY, USA, Nov. 2010, pp. 1–12, doi: 10.1145/1921168.1921203.

[9] N. Baccour, A. Koubâa, L. Mottola, M. A. Zúñiga, H. Youssef, C. A. Boano, and M. Alves, "Radio link quality estimation in wireless sensor networks: A survey," *ACM Trans. Sensor Netw.*, vol. 8, no. 4, pp. 1–33, Sep. 2012, doi: 10.1145/2240116.2240123.

[10] M. Kirubasri and N. U. Maheswari, "A study on hardware and software link quality metrics for wireless multimedia sensor networks," *Int. J. Adv. Netw. Appl.*, vol. 8, no. 3, pp. 3103–3109, 2016.

[11] S. Surendran, M. V. Ramesh, A. Montresor, and M. J. Montag, "Link characterization and edge-centric predictive modeling in an ocean network," *IEEE Access*, vol. 11, pp. 5031–5046, 2023.

[12] S. J. Olickal and R. Jose, "LSTM projected layer neural network-based signal estimation and channel state estimator for OFDM wireless communication systems," *AIMS Electron. Electr. Eng.*, vol. 7, no. 2, pp. 187–195, 2023. [Online]. Available: https://www.aimspress.com/article/doi/10.3934/electreng.2023011

[13] T. Liu and A. E. Cerpa, "Foresee (4C): Wireless link prediction using link features," in *Proc. 10th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, Apr. 2011, pp. 294–305.

[14] M. L. F. Sindjoung and P. Minet, "Wireless link quality prediction in IoT networks," in *Proc. 8th Int. Conf. Perform. Eval. Model. Wired Wireless Netw. (PEMWN)*, Paris, France, Nov. 2019, pp. 1–6. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02432805

[15] C. Boucetta, B. Nour, A. Cusin, and H. Moungla, "QoS in IoT networks based on link quality prediction," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2021, pp. 1–6.

[16] N. Diouf, M. Ndong, D. Diop, K. Talla, M. Sarr, and A. C. Beye, "Channel quality prediction in 5G LTE small cell mobile network using deep learning," in *Proc. 9th Int. Conf. Soft Comput. Mach. Intell. (ISCMI)*, Nov. 2022, pp. 15–20.

[17] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[18] M. Nsaif, G. Kovásznai, A. Malik, and R. de Fréin, "SM-FPLF: link-state prediction for software-defined DCN power optimization," *IEEE Access*, vol. 12, pp. 79496–79518, 2024.

[19] Q. Wang, J. Fang, B. Gong, X. Du, and M. Guizani, "An intelligent data uploading selection mechanism for offloading uplink traffic of cellular networks," *Sensors*, vol. 20, no. 21, p. 6287, Nov. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/21/6287

[20] J. Lee, Y. Yi, S. Chong, and Y. Jin, "Economics of WiFi offloading: Trading delay for cellular capacity," *IEEE Trans. Wireless Commun.*, vol. 13, no. 3, pp. 1540–1554, Mar. 2014.

[21] D. Saliba, R. Imad, S. Houcke, and B. E. Hassan, "WiFi dimensioning to offload LTE in 5G networks," in *Proc. IEEE 9th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2019, pp. 0521–0526.

[22] S. Deng, A. Sivaraman, and H. Balakrishnan, "All your network are belong to us: A transport framework for mobile network selection," in *Proc. 15th Workshop Mobile Comput. Syst. Appl.*, New York, NY, USA, Feb. 2014, pp. 1–6, doi: 10.1145/2565585.2565588.

[23] S. Anbalagan, D. Kumar, D. Ghosal, G. Raja, and M. V, "SDN-assisted learning approach for data offloading in 5G HetNets," *Mobile Netw. Appl.*, vol. 22, no. 4, pp. 771–782, Aug. 2017.

[24] A. Ford and C. Raiciu, *TCP Extensions for Multipath Operation With Multiple Addresses*, Standard 6824, RFC Editor, Jan. 2013. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-ietf-mptcp-multiaddressed-12

[25] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure, "Exploring mobile/WiFi handover with multipath TCP," in *Proc. ACM SIGCOMM Workshop Cellular Netw., Oper., Challenges, Future Design*, New York, NY, USA, Aug. 2012, pp. 31–36, doi: 10.1145/2342468.2342476.

[26] F. Biersack, K. Holzinger, H. Stubbe, T. Wild, G. Carle, and A. Herkersdorf, "Priority-aware inter-server receive side scaling," in *Proc. 31st Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, Mar. 2023, pp. 51–58.

[27] P. Du, Q. Zhao, and M. Gerla, "A software defined multi-path traffic offloading system for heterogeneous LTE-WiFi networks," in *Proc. IEEE 20th Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2019, pp. 1–9.

[28] J. Ling, S. Kanugovi, O. Marce, and S. Vasudevan, "Performance gains of a hybrid Wi-Fi/LTE architecture," in *Proc. IEEE 81st Veh. Technol. Conf. (VTC Spring)*, May 2015, pp. 1–6.

[29] C. Chen, L. Jiang, S. Jun, H. Zhu, J. Gu, and X. Chang, "A machine learning based mptcp subflow control algorithm for multi-access heterogeneous networks," in *Proc. IEEE 22nd Int. Conf. Commun. Technol. (ICCT)*, 2022, pp. 1817–1822.

[30] C. Hopps, "RFC2992: Analysis of an equal-cost multi-path algorithm," IETF, Request Comments (RFC), USA, Tech. Rep. 2992, 2000. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc2992

[31] (2021). *Scikit-learn*. [Online]. Available: https://scikit-learn.org/stable/

[32] (2021). *Keras Simple. Flexible. Powerful*. [Online]. Available: https://keras.io/

[33] M. Singh, N. Varyani, J. Singh, and K. Haribabu, "Estimation of end-to-end available bandwidth and link capacity in sdn," in *Ubiquitous Communications and Network Computing*, N. Kumar and A. Thakre, Eds., Cham, Switzerland: Springer, 2018, pp. 130–141.

[34] S. Kamath, A. Srivastava, P. Kamath, S. Singh, and M. S. Kumar, "Application aware multiple constraint optimal paths for transport network using SDN," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 4, pp. 4376–4390, Dec. 2021.

[35] K.-K. Yap, R. Sherwood, M. Kobayashi, T.-Y. Huang, M. Chan, N. Handigol, N. McKeown, and G. Parulkar, "Blueprint for introducing innovation into wireless mobile networks," in *Proc. 2nd ACM SIGCOMM Workshop Virtualized Infrastruct. Syst. Archit. (VISA)*, New York, NY, USA, Sep. 2010, pp. 25–32, doi: doi.org/10.1145/1851399.1851404.

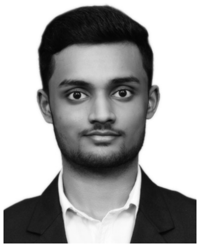
[36] K. Schneider, *Universal Mobile Telecommunication System Long Term Evolution (UMTS-LTE)*. Hoboken, NJ, USA: Wiley, 2011, ch. 1, pp. 409–488. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119970460.ch14

[37] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 2, pp. 51–62, Mar. 2007, doi: 10.1145/1232919.1232925.

[38] PICA8. (2019). *Creating a Group Table*. [Online]. Available: https://docs.pica8.com/display/PICOS2111cg/Creating

[39] G. F. Networks, "4G/5G Network experience evaluation guideline," GSMA, London, U.K., Tech. Rep. Feb. 2, 2020. [Online]. Available: https://www.gsma.com/solutions-and-impact/technologies/networks/wp-content/uploads/2020/02/4G5G-Network-Experience-Evaluation-Guideline-_GSMA.pdf

SANTHOSHA KAMATH received the B.E. degree from Vishweshwarayya University Belgaum, Karnataka, India, in 2004, the M.Tech. degree in network engineering from Manipal University, in 2009, and the Ph.D. degree in software-defined networking from Manipal Institute of Technology, MAHE, Manipal, India. His research interests include SDN, OTT applications, embedded system designs, cognitive networks, and ad-hoc networks.



J. ARAVINDA RAMAN received the B.Tech. degree in information technology from Manipal Institute of Technology, Manipal, India, in 2023. He is currently pursuing the Master of Computer Science degree with North Carolina State University, Raleigh, NC, USA. He is a Data Science Intern with The Masen Group, LLC, Syracuse, NY, USA. His research interests include machine learning, natural language processing, and graph neural networks.

PANKAJ KUMAR (Member, IEEE) received the B.Tech. and M.Tech. degrees (Hons.) from Uttar Pradesh Technical University, Lucknow, India, and the Ph.D. degree from Indian Institute of Technology (IIT) Ropar, India. He was a Postdoctoral Fellow with Indian Institute of Science (IISc), Bengaluru, India. He is currently an Assistant Professor with the Department of Information and Communication Technology, Manipal Institute of Technology, Manipal, India. His research interests include cooperative networks, network coding, network-coded cooperation, channel correlation, energy harvesting, VANET, ad hoc networks, and NOMA.

SANJAY SINGH (Senior Member, IEEE) received the degree in electronics and telecommunications from the Institution of Electronics and Telecommunications Engineers, New Delhi, India, in 2001, and the M.Tech. and Ph.D. degrees from Manipal Institute of Technology, Manipal, India, in 2003 and 2010, respectively. Since 2004, he has been with the Department of Information and Communication Technology, Manipal Institute of Technology, MAHE, where he is currently a Professor. His research interests include artificial intelligence, machine learning, neural networks, fuzzy logic, and natural language processing. He is a Senior Member of ACM.

M. SATHISH KUMAR (Member, IEEE) received the degree in electronics and communication engineering from the Institution of Engineers, India, the master's degree in engineering specializing in optical communication from the College of Engineering, Guindy, Anna University, and the Ph.D. degree from the National Institute of Technology Karnataka (NITK), Surathkal, India. From 2008 to 2010, he was a Research Assistant Professor with the School of Electrical Engineering and Computer Science, Seoul National University, South Korea. Since 2010, he has been a Professor with the Electronics and Communication Engineering Department. He works in wireless communications, optical communications, and nanophotonics.