World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:17, No:1, 2023

Traffic Forecasting for Open Radio Access Networks Virtualized Network Functions in 5G Networks

Khalid Ali, Manar Jammal

***Abstract*—**In order to meet the stringent latency and reliability requirements of the upcoming 5G networks, Open Radio Access Networks (O-RAN) have been proposed. The virtualization of O-RAN has allowed it to be treated as a Network Function Virtualization (NFV) architecture, while its components are considered Virtualized Network Functions (VNFs). Hence, intelligent Machine Learning (ML) based solutions can be utilized to apply different resource management and allocation techniques on O-RAN. However, intelligently allocating resources for O-RAN VNFs can prove challenging due to the dynamicity of traffic in mobile networks. Network providers need to dynamically scale the allocated resources in response to the incoming traffic. Elastically allocating resources can provide a higher level of flexibility in the network in addition to reducing the OPerational EXpenditure (OPEX) and increasing the resources utilization. Most of the existing elastic solutions are reactive in nature, despite the fact that proactive approaches are more agile since they scale instances ahead of time by predicting the incoming traffic. In this work, we propose and evaluate traffic forecasting models based on the ML algorithm. The algorithms aim at predicting future O-RAN traffic by using previous traffic data. Detailed analysis of the traffic data was carried out to validate the quality and applicability of the traffic dataset. Hence, two ML models were proposed and evaluated based on their prediction capabilities.

***Keywords*—**O-RAN, traffic forecasting, NFV, ARIMA, LSTM, elasticity.

I. Introduction

THE ever-increasing demand in the mobile networks has put a significant burden on the current mobile network infrastructure. The 5th Generation of mobile communications (5G) is expected to support a variety of services and applications by having a more stringent latency requirement and consuming a significantly higher bandwidth compared to the previous generations. However, this level of performance comes at the cost of having a more extreme requirements in terms of latency, reliability, computation power, bandwidth, etc. for the underlying network. O-RAN has been proposed to address many of the limitations of the current RAN through incorporating Cloud and Intelligence in the existing architecture. Due to O-RAN virtualization combined with its latency-aware architecture, O-RAN enables supplier diversity, reduces the deployment and maintenance cost, and exploits state-of-the-art intelligent solutions for network optimization and healing [1]. Telecom operators provision a vast number of resources which have a significant operational and maintenance costs, in addition to its inflexibility in terms of scalability [2]. Hence, they are deemed impractical and inefficient regarding the 5G application architectures. NFV has removed the limitations on network evolution through decoupling the network functions from the network appliances and deploying them on a commercial off-the-shelf hardware [3]. Thus, reducing the OPerational and CApital EXpenditure (OPEX/CAPEX) [4]; providers have registered 49% CAPEX savings compared to traditional deployment [5]. Additionally, it enables exploiting state-of-the-art ML solutions that reduces OPEX. OPEX is modeled based on the costs associated with operating the VNFs; it consists of the cost of running a VNF, deploying a VNF, backup VNFs, and forwarding the traffic. To reduce these costs resource management and allocation techniques are employed. Concerning resource allocation, the research problems that arise are the server placement, function placement, and dynamic resource management. The server and function placement problems are explored thoroughly in the literature [3], [7], [10], [16]. In contrast, dynamic resource management problem in NFV is less saturated. Dynamic resource management can be defined as dynamically scaling the resources allocated for VNFs in accordance with the real-time network demand [6]. In a nutshell, the more resources allocated for a VNF in a Service Function Chain (SFC), the more traffic that VNF can handle. However, over-provisioning resources can result in a low utilization level due to traffic dynamicity. Hence, over-provisioning can result in increasing the cost of those VNFs. Systems that dynamically allocate resources according to the traffic are called elastic. Elasticity is defined as the degree to which a system can adapt to traffic changes by provisioning and deprovisioning resources in an autonomic manner [7].

In the context of O-RAN, applying the technology of NFV has many benefits. The O-RAN Alliance assumes O-RAN components (Distributed Unit (DU), Control Unit (CU), and the Near-Real Time RAN Intelligent Controller (Near-RT RIC)) to be considered as VNFs; hence, allowing the exploitation of ML solutions in O-RAN [8]. Through applying intelligent ML based elastic policy in O-RAN, cost of running those VNFs can be reduced, making it more sustainable and agile while maintaining Service-Level Agreement (SLA).

To this end, several elastic models have been developed in the literature aiming to elastically scale resources (e.g., in [2], [6], [11]-[14]). However, only a few have considered a proactive approach, as opposed to a reactive one. Reactive models scale the VNF according to arrived demand, which can result in a significant delay stemming from copying the VM image and

Khalid Ali and Manar Jammal are with School of Information Technology, York University, Toronto, Canada (e-mail: kkali@yorku.ca, mjammal@yorku.ca).

instantiating new VNF instances. This delay can violate SLA [6]. To address this, an ML based proactive approach is proposed. Simply put, if one can successfully predict the traffic, new VNF deployment can be done in advance, thus rendering the deployment delay irrelevant to maintaining the SLA [2].

The remainder of this paper is organized as follows: Section II presents an overview of the related work in the literature. Section III thoroughly explores the problem at hand and discusses the mathematical system model involved. Section IV introduces the proposed traffic prediction model. Section V presents the performance evaluation of the implemented models. Finally, Section VI summarizes the work done and discusses future work.

II. Related Work

Recently, dynamic resource allocation has been studied in the context of NFV in general and the RAN in particular. Many algorithms in the literature are concerned with optimizing the resource allocation. Bari et al. formulated this problem into an Integer Linear Programming (ILP) and came up with a heuristic solution for large scale networks [1]. Yuan et al. presented a pooling deployment approach that achieved a fine-grained management of the resources [9]. Cohen et al. investigated VNF geo-placement over different datacenters while minimizing the cost [10]. However, they deal with a static model, not considering the dynamicity of traffic. Elasticity models consider this variability in behavior. Regarding VNF scaling, Arteaga et al. presented a VNF adaptive scaling using Reinforcement Learning (RL), however, the model was SFC specific and does not work with different SFCs [11]. Wang et al. proposed a dynamic instance provisioning model for enterprise services [12]. The model takes in account the traffic and the server capacity; however, it is reactive; it can result in SLA violation. In [13] and [14], the models developed were proactive in nature; they predict the traffic and scale accordingly. Bilal et al. formulated the problem into a timeseries one to predict resource usage, thus achieving an elastic NFV [15]. Clayman et al. focused on developing a dynamic placement model that can handle increasing demand by installing virtual routers, however, it did not have a deprovisioning mechanism [16]. Cloud Providers (CPs) implement elastic solution as additional services. Amazon Auto Scaling Group and Microsoft Azure offer the tenants horizontal scaling solutions [17], [18]. CloudScale and PRESS have employed vertical scaling solutions for allocating and releasing resources, however, in most cases it does not support changing the resources on-the-fly [19], [20]. Thus, vertical scaling is not recommended by CPs [21]. A latency-aware scaling solution requires scaling decision to be predicted [22]. Some works have employed ML models for prediction, Mijumbi et al. employed Graph Neural Network (GNN) to model topological dependencies of VNF Chains [23], [24]. It performed well compared to conventional scaling models, however, the accuracy drops when testing on new data implying a low generalization accuracy.

In the context of the 5G O-RAN, resource utilization can be maximized through applying the elasticity techniques of scaling, be it reactive or proactive. Several developed solutions in the literature focus on elasticity in 5G O-RAN. For instance, Sarrigiannis et al. proposed an intelligent solution for NFV orchestration consisting of a latency-aware placement in addition to an online scheduling algorithm that elastically scales VNFs in a 5G architecture [25]. However, scaling is done reactively resulting in a delay associated with deploying new instances. Gutierrez-Estevez et al. have introduced a reactive ML based Elastic Resource Management Model [26]. CPs such as Amazon have started introducing elastic solutions in the context of 5G networks [17]. However, these solutions are costly and reactive [27].

To the best of our knowledge, a proactive elastic model for resource allocation in the context of 5G O-RAN investigated in this work has not been studied in the literature. Most existing work is reactive. Moreover, the existing proactive models are generic and do not consider 5G specific requirements. In fact, elasticity in O-RAN environment is still an unexplored area of research. Hence, in this work we aim to address this gap by designing and implementing a proactive Elastic VNF orchestration policy for 5G O-RAN.

III. Problem Statement

Form a CP perspective, a traffic request in O-RAN can be modeled as traffic originating from a source (O-RU) that is propagated through the O-RAN SFC consisting of the O-DU, O-CU, Near-RT RIC (Fig. 1). Elastic VNF orchestration policy with the objective of minimizing the OPEX through scaling VNF instances in a proactive manner consists of two subproblems; when to scale (traffic prediction) and how to scale (scaling decision and provisioning). The focus of this work is the former traffic prediction problem. As discussed, several advanced ML model are suited for traffic prediction. However, their performance is dependent on the utilized data. Since the amount of deployed O-RAN VNFs are determined by the time-varying traffic, it is crucial to obtain a prediction with minimal error. In this section, an overview of the O-RAN environment is given, highlighting the crucial operational aspects. Moreover, the problem of traffic forecasting is formulated with the focus on the important aspects ensuring accurate predictions.

A. O-RAN Overview

RAN are a major component of wireless communication systems. The role of RANs is to connect the User Equipment (UE) to the core network. Traditionally, it is done with no regards to the application services. O-RAN considers the application adding agility to the network. However, the more layers added, the higher the latency and the more computational power required, thus, increasing the complexity. To overcome these challenges modern learning techniques and MEC capabilities are employed in O-RAN infrastructure [28]. The distribution of functionality in the O-RAN increases the reliability by avoiding a single node failure. The separation of the control plane and the user plane enables implementation on a server platform and since they are intended to work separately, it allows for scalability and increases flexibility in the O-RAN. Simply put, the O-RAN is a virtualized, application specific, and software-oriented RAN enabling it to support high speed

applications and IoT network [29]. The O-RAN architecture is based on decoupling the non-real time functionalities and the real time one, introducing RIC. RICs host ML functionalities; the service and model training hosted in the non-RT RIC and the trained models hosted in the near-RT RIC. The near-RT RIC contains the databases that tracks the performance of the network through the E2 and A1 interfaces. The E2 interface relays metrics from CU and DU to near-RT RIC, which in turn provides orchestration and management using ML. The A1 interface relays the ML policies and training models to the non-RT RIC. The non-RT RIC supports resource management and provides guidance to the near-RT RIC [30]. Additionally, O-RAN virtualizes its components; DU, CU, Near-RT RIC are VNFs (Fig. 1) [4]. Moreover, O-RAN uses ML techniques to develop smarter RAN layers in its architecture [3]. All those factors combined paved the way for a new direction in research applying intelligent ML based VNF orchestration techniques on RANs, addressing existing issues.

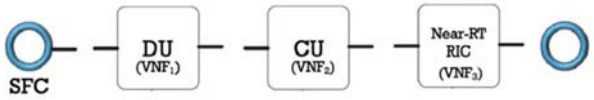

Fig. 1 O-RAN SFC

B. Traffic Forecasting

Traffic forecasting is a crucial part of the elastic orchestration problem. Predicting traffic accurately enables the elastic model to closely match the resources to the traffic demands ahead of time, thus reducing the operating cost and increasing the utilization as discussed previously in Section I. Traffic forecasting can be modeled as a timeseries forecasting problem. Timeseries forecasting can be formulated as a supervised learning problem where the features input to the model are the past observations (Fig. 2). Therefore, the prime objective is to use past observations to predict the traffic. However, what makes timeseries forecasting problems unique is the dependency between the features, i.e., the model is used to predict a quantity based on a lagged version of the quantity itself. Moreover, prediction confidence diminishes as the model extrapolates further into the future. Therefore, it is inherently more challenging to predict values in a time series as opposed to a regular regression problem. The forecasting problem can be formulated as:

$$\alpha^*(t+m) = f\big(\alpha(t), \alpha(t-1), \alpha(t-2), \dots, \alpha(t-n)\big) \quad (1)$$

where α^* is the predicted traffic m timesteps in the future, α is the actual observed traffic, and n is the size of the observation window.

Since the prediction depends entirely on the past observations, it is of the utmost importance to ensure the quality of the data before going ahead with model implementation. The quality of the data is dictated by two main factors: the time length of the observations and the predictability of the series itself. Firstly, the length of the available series matters greatly in training the model as the available number of samples must be greater than the number of model parameters [31]. Hence, the more complex the model is, the more samples it needs for training. Secondly, the predictability of the series quantifies the regularity and the predictability of the fluctuations in the series; the more regular and repeated patterns the series has, the easier it is to forecast. The predictability depends on the temporal granularity of the of the observations; temporal aggregation significantly degrades the predictability of the series. The predictability of the series can be ensured through applying some timeseries analysis techniques. By analyzing the components of the series through decomposition, the predictability can be inferred. Any timeseries can be decomposed into three main components: i) a trend component, representing the traffic growth over time, ii) a seasonal component, representing the cyclical behavior of the series, and iii) a remainder component, representing abnormalities due to sudden events (such as cell outage in the case of RAN). For a series to be predictable it must have a high level of stationarity, meaning the statistical properties of the series (mean, variance, and autocorrelation) are not a function of time. Moreover, the parameters for the model can be deduced for the statistical properties of the timeseries. For instance, the size of the observation window is inferred from the autocorrelation function. Also, depending on the model used some transformations maybe needed to ensure stationarity or to ease the prediction process.

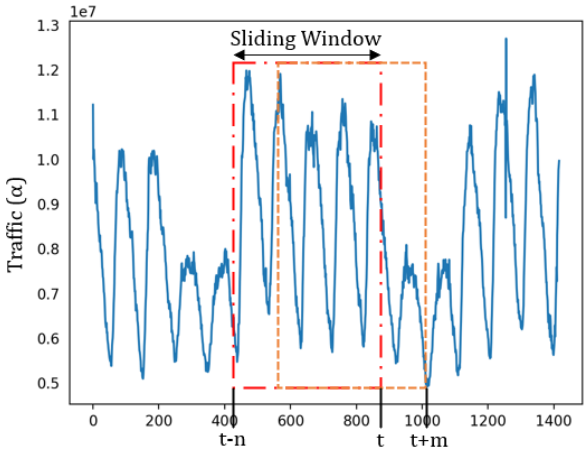

Fig. 2 Sliding window for time series prediction

IV. Proposed Solution

The traffic forecasting problem is modeled as a supervised learning problem where the task is to predict the upcoming traffic according to the features extracted from a sliding window. In other words, the model learns the pattern between the provided features and the labeled predictions. Hence, the first step is to acquire the traffic data, apply some preprocessing to extract the features from the sliding window, then feed the features to the model. By acquiring a prediction from the model, the elastic orchestrator can use it to compute the number of O-RAN VNFs required to handle the traffic. By comparing the required VNFs with the current provisioned one, a scaling decision can be reached. Finally, the scaling decision is relayed to the RIC for the deployment. In this section, the dataset and its

features are first discussed along with some analysis to ensure the stationarity and predictability of the series. Afterward, ML models are discussed along with model specific transformations and preprocessing.

A. Traffic Data

As previously mentioned, the quality of the data utilized for training and testing is crucial to the performance of the model. There are two measures that show how suitable a time series for prediction: the autocorrelation function (ACF) and the Approximate Entropy of the timeseries. The ACF shows how correlated the series is with lagged versions of itself; the higher the correlation the easier it is to predict. Approximate Entropy is used to quantify the regularity and predictability of fluctuations in a timeseries; the higher the approximate entropy, the more difficult it is to forecast it. Three datasets (Internet2 [1], Geant Network [7], Italia Telecom [32]) have been examined to determine their predictability and their applicability based on the mentioned measures. All three datasets contain traffic matrices for representing the mobile traffic data between pairs of nodes for different timestamps. The Internet2 has a temporal granularity of 5-minutes, while the Italia Telecom has 10-minuites, and the Geant has 15-minuites. The scatter plot shown in Fig. 3 shows a comparison between the ACFs of the three datasets using lags values 1 to 4. Expectedly, as the lag value increases the timeseries becomes less correlated. However, it is noticeable that the Italia dataset and the Geant network datasets are still correlated for a higher value for the lag. Table I shows the Approximate Entropy for each one. After a careful analysis the Italia Telecom dataset proved to be the most suitable of the three.

The Italia dataset contains two sub datasets; Milano and Trento; each contains a real-life call detail records collected for billing purposes in both provinces. Trento dataset includes 11466 cells while Milano has 10000 cells distributed to cover the entire area. The data are collected in over 10 minutes intervals from 31 October 2013 up until 1 January 2014. Each timestep contains the following features:

- SMS-in,
- SMS-out,
- Call-in,
- Call-out,
- Internet Traffic Activity.

Fig. 4 shows a sample of the traffic through cell number 10000 in Trento. Table I shows a description of the traffic trace in Trento and Milano sub dataset.

TABLE I
APPROXIMATE ENTROPY FOR EACH DATASET

	Internet2	Geant	Italia
Approximate Entropy	1.346	0.375	0.196

The traffic of a single cell is difficult to predict due to many external events that influence the fluctuations in that particular cell. These fluctuations cause the series to have a lower autocorrelation, thus decreasing the value of past knowledge in the prediction. To overcome this challenge, we aggregate the traffic spatially over the cells, reducing the influence of individual events happening in cells and increasing the autocorrelation through maximizing the daily patterns. The scatter plot shown in Fig. 5 shows a comparison between the autocorrelations of the individual cells vs. that of the aggregated series for lag 1 through to lag 4. It is evident that the aggregated series is more correlated with itself for different lags. Fig. 6 shows the trace for the aggregated series. As expected, the aggregation has reduced the fluctuation and magnified the cyclical patterns.

The aggregated series has an Approximate Entropy of 0.196, while the Approximate Entropy for cell 10000 is 1.386. Augmented Dickey Fuller (ADF) test was used to check for stationarity of the series. The ADF test showed the series to be stationary. Overall, the aggregated timeseries of the Italia dataset shows promising results for the predictability of the timeseries.

TABLE II
STATISTICAL DESCRIPTION FOR THE AGGREGATED DATA IN TRENTO AND MILANO

	Trento	Milano
Count	8928	8928
Mean	95872.12	621964
Standard Deviation	37812.2	223385
Minimum	39934.18	20811
Lower than 25%	65270.7	413368
Lower than 50%	98718.65	64178
Lower than 75%	111219.2	810414
Maximum	395858.67	1234958

B. ML Models

In this part, the proposed ML models are discussed. Using the Italia dataset, an ML model is to be trained to acquire the traffic prediction. After considering the available models, two ML models were chosen: Auto Regressive Integrated Moving Average (ARIMA), classical statistical model, and Long-Short Term Memory (LSTM) as a state-of-the-art algorithm. These models have been chosen for their superior performance in timeseries forecasting problems [33]. A brief description of the models and their setup is discussed below:

ARIMA

It is a class of statistical models for analyzing and forecasting timeseries data. It can represent a given timeseries based on its own past values, i.e., its own lags and the lagged forecast errors. ARIMA process can be described as ARIMA (p, d, q); the parameters p, d, and q describe the non-seasonal part of the timeseries, where p is the order of the auto-regression, d is the level of difference, and q is the order of the moving average. Mathematically, ARIMA model can be represented as:

$$\alpha_t = A + B_1\alpha_{t-1} + B_2\alpha_{t-2} + \ \dots \ + B_p\alpha_{t-p}\epsilon_t + \phi_1\epsilon_{t-1} + \phi_2\epsilon_{t-2} + \ \dots \ + \phi_q\epsilon_{t-q} \quad (2)$$

In designing the ARIMA model the objective is to determine the values of p, d, and q. Firstly, the value of d dictates the number of times the series is differenced, which in turns ensures the stationarity of the series by eliminating the trend. To deduce the value of d, the series is checked for stationarity using the ADF test after each round of differencing. The number of times

the series has had to be differenced to become stationary represents the value of d. Secondly, the value of p can be determined through the Partial Autocorrelation Function (PACF). By plotting the PACF, the number of lags over the significance line dictates the value of p. Lastly, similar to p, the value of q is taken from the plot of the ACF. Additionally, prior to inputting the series to the model some preprocessing is required to ensure the stationarity of the series and its compatibility with the model. As discussed previously, the trend is eliminated through differencing, however the seasonality is not. Therefore, it must be eliminated beforehand. The seasonality is eliminated through a nonlinear transformation that guarantees the statistical properties of the series is not time dependent. Afterward the series is normalized and input to the model. Fig. 7 shows the new series after the transformation. The parameters for the model were chosen based on the ACF and PACF shown in Fig. 8. The values of p, d, and q were chosen to be 6, 1, and 1 respectively.

LSTM

It is a special kind of Recurrent Neural Networks (RNN), capable of learning long-term dependencies. LSTMs have an advantage over the conventional RNNs in that they are designed to avoid the long-term dependency problem. All this makes LSTM networks well-suited to make predictions based on timeseries data. Another advantage LSTMs have over RNNs is its ability to deal with the vanishing gradient problem during the training. Prior to inputting the series to the model, preprocessing in terms of sliding window and feature extraction is applied to reshape the dataset from a sequential dataset to a supervised learning dataset with the present and past N values as the features and the value at $t+M$ as the label. Since the value of the autocorrelation in the dataset gets lower as the lag increases, thus reducing the accuracy of the prediction, only one timestamp prediction was chosen as the goal to ensure the accuracy. Therefore, the value of M is set to 1. As for N, the ACF shows that the last somewhat significant correlation is at 11 lags, thus signifying that adding more lags will not contribute to the accuracy of the prediction, to the contrary it will just increase the model complexity and training time. Hence, the value of N is set to 11. Moreover, two more features were added to capture the periodicity in the series and the effects of seasonality on the prediction: Day of week to make a distinction between weekday traffic vs. weekend traffic, Hour of day to differentiate between the daytime opposed to nighttime. The Granger causality test was used to ensure the usability of those features in predicting the timeseries. Finally, the dataset was rescaled to fit the criteria of the neural network. Regarding the model parameters, the evaluated model has two layers containing 100 LSTM units in each layer with a learning rate of 0.0001 using the Adam optimizer and a ReLu activation function. Glorot Uniform initializer was used for weight initialization as opposed to random initializers. The data were split into a 25% testing set and 75% training set and was fed to the model using in batches of size 16. The loss function used was the mean absolute error.

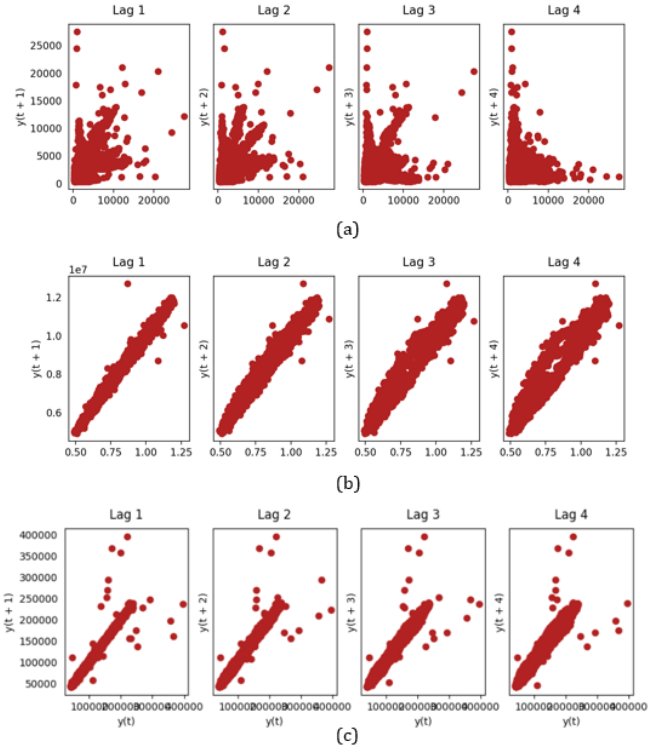

Fig. 3 Lag plot for the autocorrelation for (a) Internet2 dataset, (b) Geant dataset, and (c) Italia dataset

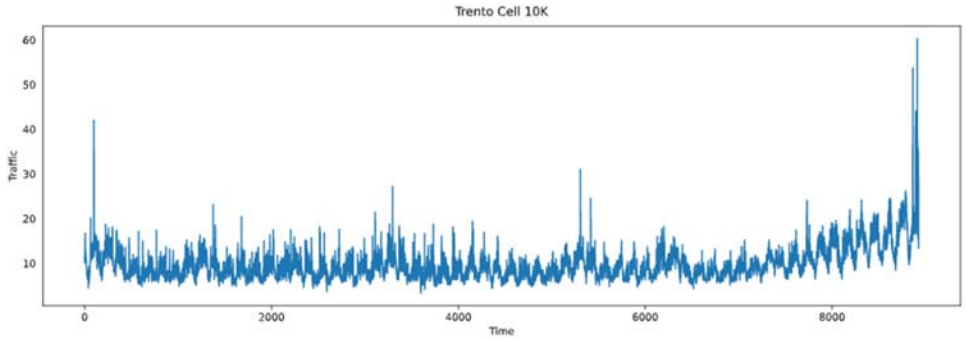

Fig. 4 Internet traffic trace for cell 10000 in Trento trace

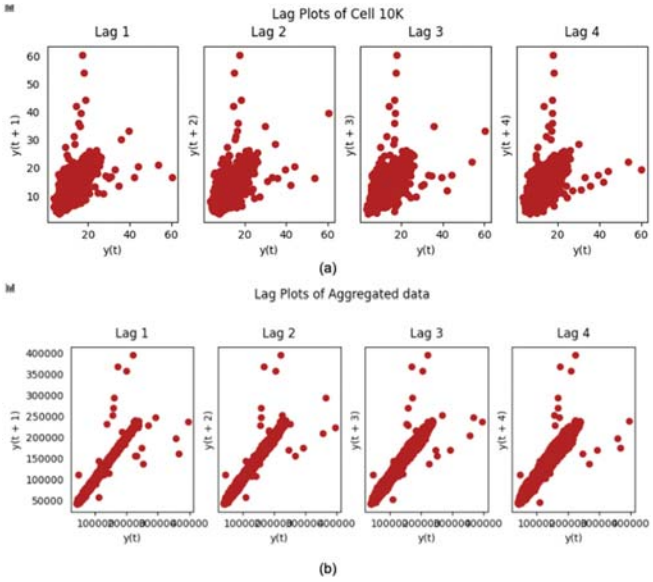

Fig. 5 Lag plot for the autocorrelation for (a) cell 10000 and (b) the aggregated data

V. Performance Evaluation

In this section, the performance of the algorithms is evaluated and compared. The evaluation is done based on the predictive capabilities of the algorithm using the traffic data from the Trento trace. Both algorithms are implemented using Python, TensorFlow, and Keras. The evaluation was performed on a computer with one Intel® Core™ i7-107000 CPU @2.90GHz, 16 GB RAM, and NIVIDIA GeForce RTX 2060 SUPER. The performance evaluation was done through mean absolute error of the internet traffic prediction vs. the actual value. Fig. 9 shows the results of the training and testing for the LSTM and Fig. 10 shows the results for the ARIMA model. For the LSTM, the training error was 2489.91 while the testing error was 2491.5. As for the ARIMA model, the testing error was found to be 3409.4 with a relatively narrow confidence area for the prediction. Both models gave a relatively low error in comparison to the mean (95872.1) and the standard deviation (37812.2) of the Trento trace.

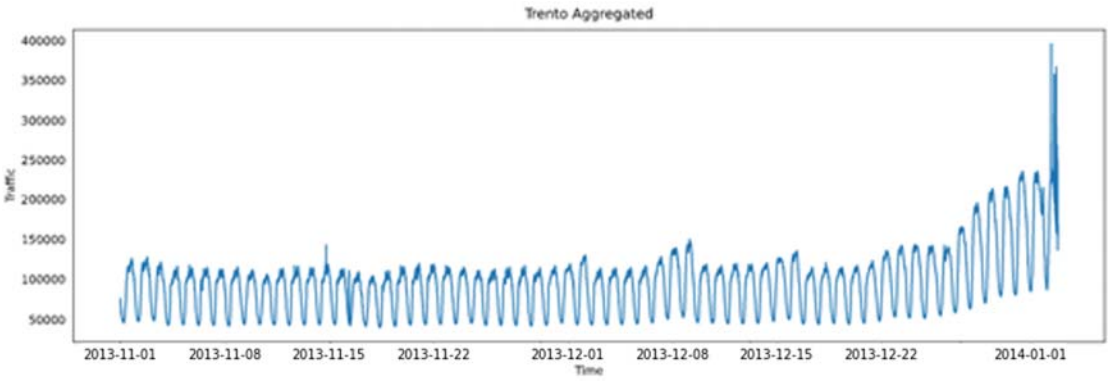

Fig. 6 Internet traffic trace for the aggregated data in Trento trace

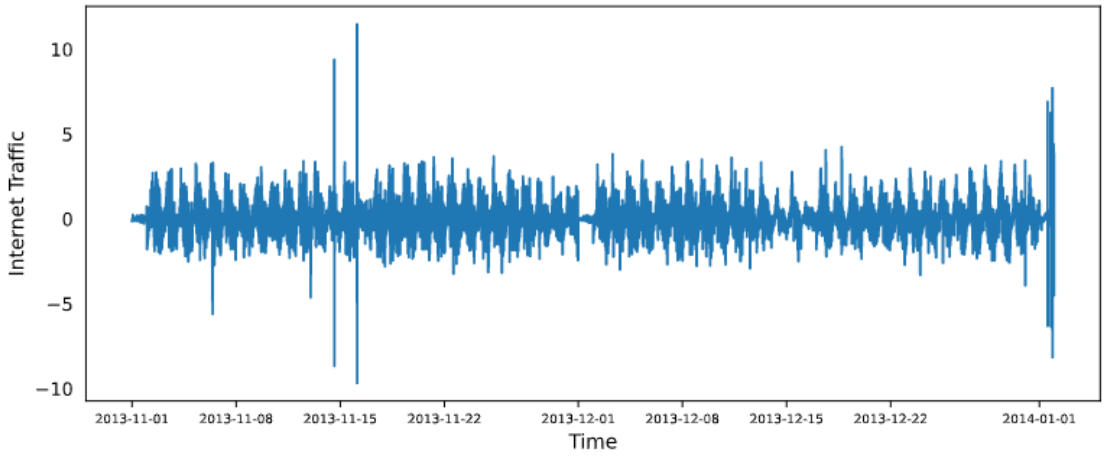

Fig. 7 Transformed series

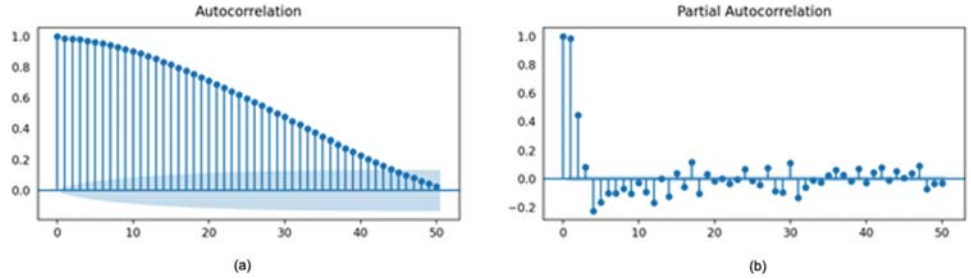

Fig. 8 The autocorrelation and PACFs for the aggregated data

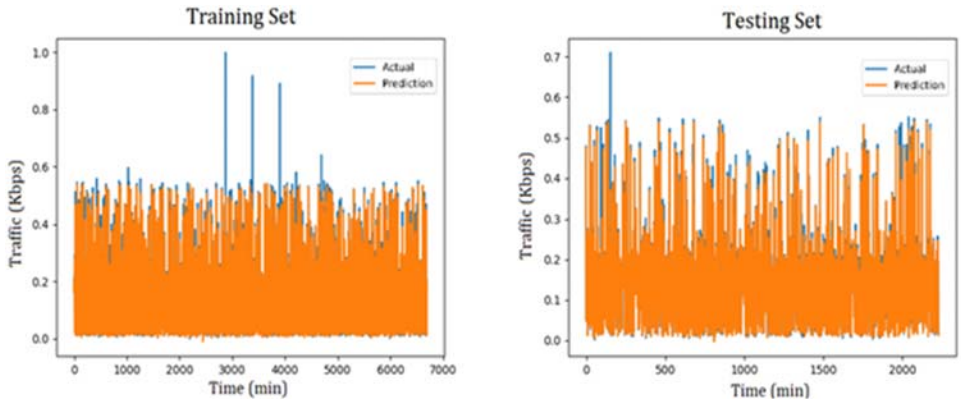

Fig. 9 Predictions vs. actual for the training and testing sets (LSTM)

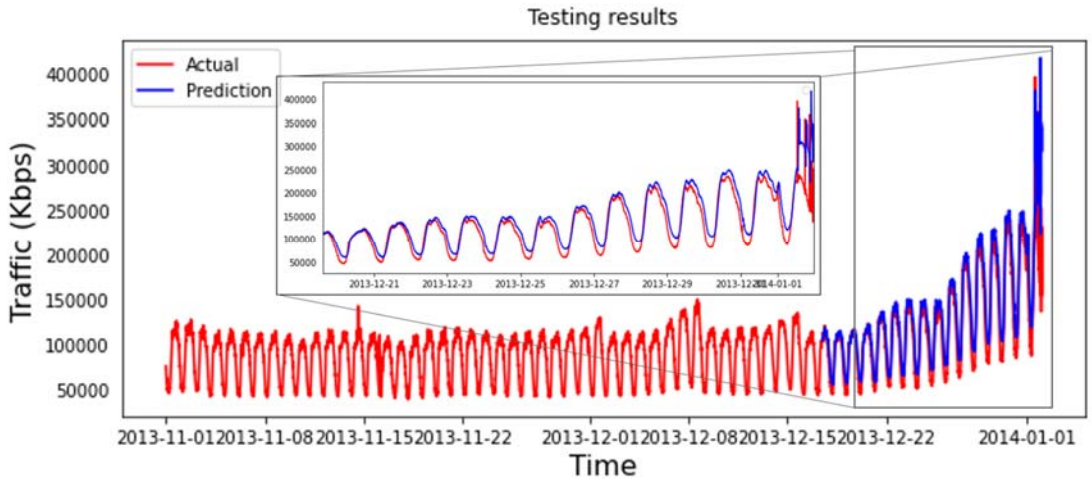

Fig. 10 Predictions vs. actual for the testing set ARIMA

Given Figs. 9 and 10 and the mentioned results, although both models gave somewhat similar performance, the LSTM slightly outperformed the ARIMA model in terms of the testing error. However, the ARIMA model responded better to the sudden spikes around the end of the trace which were caused by the surge in users during the holidays at the end of the year. The ARIMA captured the trend of the series better, but it can be seen (Fig. 10) that the prediction confidence region grows slightly wider whenever there is a sudden spike. This is expected as the LSTM captured the general patterns during the training, while the ARIMA builds the model at each time step therefore it can gradually learn that kind of anomalies. Although ARIMA performed better in terms of following the trend of the traffic, its execution time is significantly higher than the LSTM. This is again due to the fact that ARIMA build the model at each time step unlike the LSTM where you train only once. This translates to a significant discrepancy in terms of the execution time between the models. The LSTM required 375 s for the training and 3.9 s for the testing. In contrast, the ARIMA required 1360 s for the whole set. Moreover, to test the generalization of the trained LSTM model, the model was tested on the Milan trace after being trained on the Trento. Fig. 11 shows the testing results. The error was found to be 22786.9, relatively low compared to the mean (621964). Hence, the low testing error on the new trace (Milan trace) implies that the model is generalizing well.

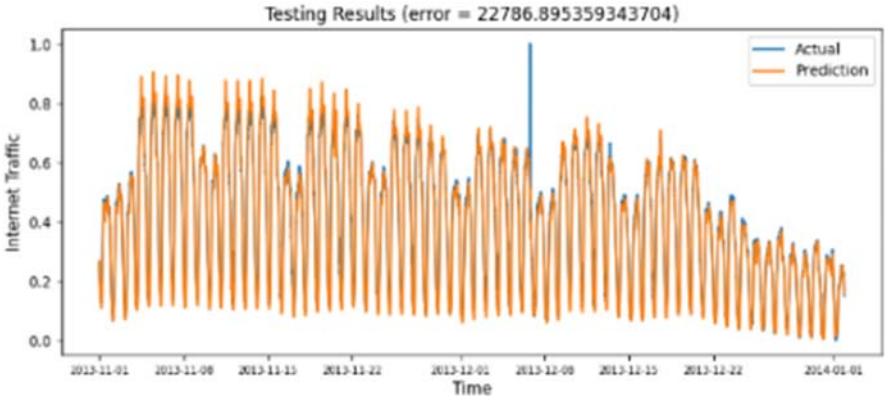

Fig. 11 Predictions vs. actual for the Milan trace LSTM

VI. Conclusion

Employing intelligent solutions for dynamic resource management is a key advantage of the O-RAN virtualization. Efficient and proactive allocation of resources is essential to ensure preserving the O-RAN requirements. As a step towards ensuring an elastic O-RAN, traffic forecasting is a crucial element. The prediction accuracy is pivotal in implementing elastic techniques aimed toward OPEX reduction. In this work, we investigated traffic forecasting in the context of O-RAN. To that end, two models (LSTM and ARIMA) were developed and implemented using Python. The models profiled and predicted the traffic based on past traffic. Moreover, we explored timeseries forecasting problem in addition to discussing different evaluation metric for the traffic dataset. The models were trained and tested using the traffic dataset provided in [32]. The performance of the models is then compared to evaluate their performance. Results have shown that the LSTM has slightly outperformed the ARIMA model in terms of overall error, while the ARIMA better modeled the spikes. According to the training nature of the models and their execution time, it can be deduced that the LSTM is better fit for this application as it has significantly lower execution time and the training is done offline, i.e., the trained model is to be hosted on the near-RT RIC where the historical traffic measurements are stored. The obtained results are promising and can form a strong foundation to further improve and advance the models. Future work will focus on improving the accuracy and extending the model to account for drastic changes in the traffic trend (new cell site, equipment failure, social events, etc.). Furthermore, to acquire a higher resolution prediction, joint prediction can be employed to predict neighboring cells traffic. It provides a good balance between coping with the noise on the cell level and a better prediction resolution.

Acknowledgment

This work is supported in part by Ciena Canada and the Ontario Centre of Excellence.

References

[1] M. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, “Orchestrating Virtualized Network Functions,” IEEE Trans. Netw. Serv. Manag., vol. 13, no. 4, pp. 725–739, 2016.

[2] Y. Gu, Y. Hu, Y. Ding, J. Lu, and J. Xie, “Elastic virtual network function orchestration policy based on workload prediction,” IEEE Access, vol. 7, pp. 96868–96878, 2019.

[3] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, “Network function virtualization: State-of-the-art and research challenges,” IEEE Commun. Surv. Tutorials, vol. 18, no. 1, pp. 236–262, 2016.

[4] 5G PPP Architecture Working Group, “View on 5G Architecture,” Version 3.0, pp. 21–470, (Online). Available: https://5g-ppp.eu/wp-content/uploads/2019/07/5G-PPP-5G-Architecture-White-Paper_v3.0_PublicConsultation.pdf, 2019.

[5] EMA, “Reducing Operational Expense (OpEx) with Virtualization and Virtual Systems Management,” (Online). https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/vmware-solution-opex-reducing-opex-wp-en.pdf, 2009.

[6] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, “Elastic virtual network function placement,” IEEE 4th Int. Conf. Cloud Networking, CloudNet, pp. 255–260, 2015

[7] A. Laghrissi and T. Taleb, “A Survey on the Placement of Virtual Resources and Virtual Network Functions,” IEEE Commun. Surv. Tutorials, vol. 21, no. 2, pp. 1409–1434, 2019

[8] O-RAN Alliance, “O-RAN Use Cases and Deployment Scenarios Towards Open and Smart RAN,” 2020.

[9] Q. Yuan, H. Tang, Y. Zhao, and X. Wang, “An approach for virtual network function deployment based on pooling in vEPC,” IEICE Trans. Commun., vol. E101B, no. 6, pp. 1398–1410, 2018

[10] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, “Near optimal placement of virtual network functions,” Proc. - IEEE INFOCOM, vol. 26, pp. 1346–1354, 2015

[11] C. H. T. Arteaga, F. Rissoi, and O. M. C. Rendon, “An adaptive scaling mechanism for managing performance variations in network functions virtualization: A case study in an NFV-based EPC,” 13th Int. Conf. Netw. Serv. Manag. CNSM, vol. 2018-Janua, pp. 1–7, 2017.

[12] X. Wang, C. Wu, F. Le, A. Liu, Z. Li, and F. Lau, “Online VNF scaling in datacenters,” IEEE Int. Conf. Cloud Comput. CLOUD, no. 1, pp. 140–147, 2017

[13] X. Fei, F. Liu, H. Xu, and H. Jin, “Adaptive VNF Scaling and Flow Routing with Proactive Demand Prediction,” Proc. - IEEE INFOCOM, pp. 486–494, 2018

[14] X. Zhang, C. Wu, Z. Li, and F. C. M. Lau, “Proactive VNF provisioning with multi-timescale cloud resources: Fusing online learning and online optimization,” Proc. - IEEE INFOCOM, 2017

[15] A. Bilal, T. Tarik, A. Vajda, and B. Miloud, “Dynamic cloud resource scheduling in virtualized 5G mobile systems,” IEEE Glob. Commun. Conf. GLOBECOM 2016 - Proc., pp. 0–5, 2016

[16] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, “The dynamic placement of virtual network functions,” IEEE/IFIP NOMS 2014 - IEEE/IFIP Netw. Oper. Manag. Symp. Manag. a Softw. Defin. World, 2014

[17] AWS, “Amazon EC2 Secure and resizable compute capacity to support virtually any workload,”. http://aws.amazon.com/ec2/, 2021.

[18] E. Hormozi, H. Hormozi, M. K. Akbari, and M. S. Javan, “Using of machine learning into cloud environment (a survey): Managing and scheduling of resources in cloud systems,” Proc. - 7th Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput. 3PGCIC, pp. 363–368, 2012.

[19] G. Brataas, E. Stav, S. Lehrig, S. Becker, G. Kopčak, and D. Huljenic, "CloudScale: Scalability management for cloud systems," ICPE - Proc. ACM/SPEC Int. Conf. Perform. Eng., pp. 335–338, 2013.
[20] Z. Gong, X. Gu, and J. Wilkes, "PRESS: PRedictive Elastic reSource Scaling for cloud systems," Proc. Int. Conf. Netw. Serv. Manag. CNSM 2010, pp. 9–16, 2010.
[21] M. Sedaghat, F. Hernandez-Rodriguez, and E. Elmroth, "A virtual machine re-packing approach to the horizontal vs. vertical elasticity trade-off for cloud autoscaling," ACM Int. Conf. Proceeding Ser., 2013.
[22] ETSI, "Network functions virtualisation- an introduction, benefits, enablers, challenges & call for action," SDN OpenFlow World Congr., no. 1, 2012.
[23] R. Mijumbi, S. Hasija, S. Davy, A. Davy, B. Jennings, and R. Boutaba, "Topology-Aware Prediction of Virtual Network Function Resource Requirements," IEEE Trans. Netw. Serv. Manag., vol. 14, no. 1, pp. 106–120, 2017.
[24] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," IEEE Trans. Neural Networks, vol. 20, no. 1, pp. 61–80, 2009.
[25] I. Sarrigiannis, K. Ramantas, E. Kartsakli, P. V. Mekikis, A. Antonopoulos, and C. Verikoukis, "Online VNF Lifecycle Management in an MEC-Enabled 5G IoT Architecture," IEEE Internet Things J., vol. 7, no. 5, pp. 4183–4194, 2020.
[26] D. M. Gutierrez-Estevez et al., "The path towards resource elasticity for 5G network architecture," IEEE Wirel. Commun. Netw. Conf. Work. WCNCW, pp. 214–219, 2018.
[27] M. A. Sharkh, Y. Xu, and E. Leyder, "CloudMach: Cloud Computing Application Performance Improvement through Machine Learning," Can. Conf. Electr. Comput. Eng., 2020.
[28] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," IEEE Commun. Surv. Tutorials, vol. 19, no. 3, pp. 1657–1681, 2017.
[29] SAMSUNG, "Open RAN - The Open Road to 5G," SAMSUNG White Paper, https://image-us.samsung.com/SamsungUS/samsungbusiness/pdfs/Open-RAN-The-Open-Road-to-5G.pdf, 2019.
[30] J. Keeney, M. Skorupski, G. Clapp, D. Kim, H. Eiselt, and R. Lovell, "O-RAN Software Community For inclusion from Release A Non Real-Time RAN Intelligent Controller (RIC non-RT)."
[31] R. Hyndman and A. Kostenko, "Minimum Sample Size Requirements for Seasonal Forecasting Models," Foresight Int. J. Appl. Forecast., no. 6, pp. 12–15, 2007.
[32] G. Barlacchi et al., "A multi-source dataset of urban life in the city of Milan and the Province of Trentino," Sci. Data, vol. 2, pp. 1–15, 2015.
[33] C. Gijón, M. Toril, S. Luna-Ramírez, M. L. Marí-Altozano, and J. M. Ruiz-Avilés, "Long-term data traffic forecasting for network dimensioning in lte with short time series," Electron., vol. 10, no. 10, 2021.