# AN ENCODER-DECODER DEEP LEARNING APPROACH FOR MULTISTEP SERVICE TRAFFIC PREDICTION

**Theodoros Theodoropoulos**
Department of Informatics and Telematics
Harokopio University
Athens, Greece, 16671
ttheod@hua.gr

**Angelos-Christos Maroudis**
Department of Informatics and Telematics
Harokopio University
Athens, Greece, 16671
it21863@hua.gr

**John Violos**
Department of Informatics and Telematics
Harokopio University
Athens, Greece, 16671
violos@hua.gr

**Konstantinos Tserpes**
Department of Informatics and Telematics
Harokopio University
Athens, Greece, 16671
tserpes@hua.gr

August 4, 2022

**ABSTRACT**

In this research paper, we compare statistical time series with Deep Learning (DL) models. We propose an encoder-decoder DL approach for multi-step traffic prediction. We examined four encoder-decoder DL architectures i) Stacked LSTMs, ii) CNN-LSTMs, iii) Bidirectional LSTM and iv) an innovative Hybrid Unidirectional-Bidirectional LSTM. We conducted experiments using a TCP trace data set with a 5 minutes time-step. We predict the number of requests, the transmitted data and the duration of the sessions with multi-steps in a range of one to five steps, which corresponds to a time window that spans 25 minutes in total. The results show that the encoder-decoder architecture provides better accuracy results in regards to predicting the traffic and the duration of the sessions.

## 1 Introduction

Distributed computing that orchestrates and manages a pool of heterogeneous computational, communication and storage resources is a prominent solution to tackle the continuously increasing generation of data and service requests [1, 2] also named traffic. Modern computing paradigms like cloud computing and edge computing are required to facilitate the high volume and velocity requirements of big data services respectively. Cloud computing has emerged as a solution to addresses the high volume of data by providing infrastructure resources in an elastic fashion in order to keep up with the fluctuation of the workload [3]. Edge computing can address the drawbacks of cloud-based solutions by moving computation physically closer to the network edge where data are generated in order to reduce latency and bandwidth between data centres and sensors [4].

For more than one decade, the network traffic prediction is used in order to estimate the amount of cloud or edge resources required by data services [5]. This approach is conducted in the context of dynamical resource management with the ability to proactively scale up or scale down in every time period. The assumption is that if we predict the amount of data and the number of requests, we can respectively replicate the virtual machines and the network functions of the data services. In this paper we don't investigate and evaluate the cloud mechanism that makes the replication because it is common practice to use widely accepted tools like Kubernetes [6] for the data services deployment, scaling, and management. Instead, we propose the traffic prediction model that triggers the resource scaling and deployment.

Generally speaking predicting the future, is a rather hard task. Thankfully, the TCP traffic presents strong auto-correlation structure [7] that makes various statistical and time series methods capable for modeling and prediction. Auto-correlation measures the relationship between a metric's current value and its past values. Multiple network

analyses have shown that the TCP traffic is characterized by repeated patterns over time. Regarding the various prediction models utilized, for more than one decade network engineers have been using statistical models like Poisson, Autoregressive–Moving-Average (ARMA), and Autoregressive Integrated Moving Average (ARIMA). The advent of DL has drastically changed the landscape of data analytics and decision making. Specifically, in the case of time series prediction, Recurrent Neural Networks (RNNs) often significantly surpass the traditional statistical forecasting models. The first stage of this research is dedicated to answering the question if RNNs are a better option when compared to statistical time series models for the network traffic prediction of data services.

The common time series models and the simple RNNs are designed to provide only one-step predictions. Multiple-step prediction models output a sequence of values of sequential time-steps. The Multiple-step prediction regarding traffic time series is extremely important because it can be utilized in order to achieve better granularity of the resource scheduling compared to one-step prediction methodologies. The resource orchestration mechanism can implement a more sophisticated real time adaptation of the intensive data driven workflows using multi-step traffic insight [8] because each virtual device and service function has different deployment time. The encoder-decoder can be used for multiple-step time series forecasting. Encoder-decoder is a composite DL architecture that consists of two Artificial Neural Networks (ANN) that interact through latent variables and makes sequence to sequence predictions. The modeling and the applicability of the encoder-decoder architectures for service traffic prediction has yet to be proposed in the literature and it is an important novelty of our research.

The four major contributions of our work are:

- We compare theoretically and experimentally well established statistical time series models to DL approaches for service traffic prediction.
- We propose the use of encoder-decoder models for multi-step service traffic prediction.
- We analyze and evaluate multiple neural network topologies for the encoder-decoder.
- To the best of our knowledge we are the first to try a unidirectional - bidirectional LSTM encoder.

The rest of the paper is structured as follows: Section 2 highlights the related work in service traffic prediction, time series and encoder-Decoders. Section 3 explains the functionality of traffic prediction in the context of big data services. Section 4 explain how the statistical time series model are used in traffic prediction. Section 5 contains the analysis of the encoder-decoders utilized and explains how they can be used for the multi-step traffic prediction. Section 6 describes the experimental setup and the evaluation results. Finally, Section 7 concludes the paper and suggest directions for future work.

## 2 Related Work

For many years different methods have been used for modeling and forecasting service traffic. In the beginning, point process statistical models like Poisson processes were used but they presented the limitation that they do not capture the self-similarity characteristic [9] of the sequence values. Afterwards, time series models such as Autoregressive–Moving-Average (ARMA) and their variations Autoregressive Integrated Moving Average (ARIMA) and Seasonal ARIMA (SARIMA) [10] were used for traffic prediction and managed to minimize the operation cost taking into account two types of cost: i) The cloud resource costs which occurs when non-essential resource provisioning is performed due to traffic overestimation and ii) The QoS degradation cost which occurs when the traffic is underestimated, resulting to fewer resources than actually needed being allocated and thus jeopardizing the satisfaction of the users of the data services.

With the advent of DL, many decision making models after being experimentally compared and redesigned, were ultimately replaced by ANN. The first studies showed that ARIMA performs better than simple feed-forward ANN [11]. The reason is that simple feed-forward ANN are not designed for sequential tasks. They allow information to travel one way and can not capture the periodic and autocorrelation patterns that characterize network traffic. RNNs is a different class of ANN that models temporal sequencing of data so that each observation is dependent on the previous ones running in both directions by loops in their network. Information derived from earlier input is fed back into the network providing a kind of memory of the previous observation sequences in order to predict the next one. Complex RNN models that leverage interactive and temporal behaviour of data centers have been used successfully for single-service traffic prediction and interactive network traffic prediction [12].

Many data transfer, storage and processing services include short and long range time dependencies, making the multi-step prediction a prominent solution [13]. Multi-step prediction using RNN with iterated prediction over many time steps has been applied for IoT traffic time series prediction [14]. This approach is based on the assumption that for each step prediction the output of the RNN goes in the input in order to make the next step prediction. The limitation is

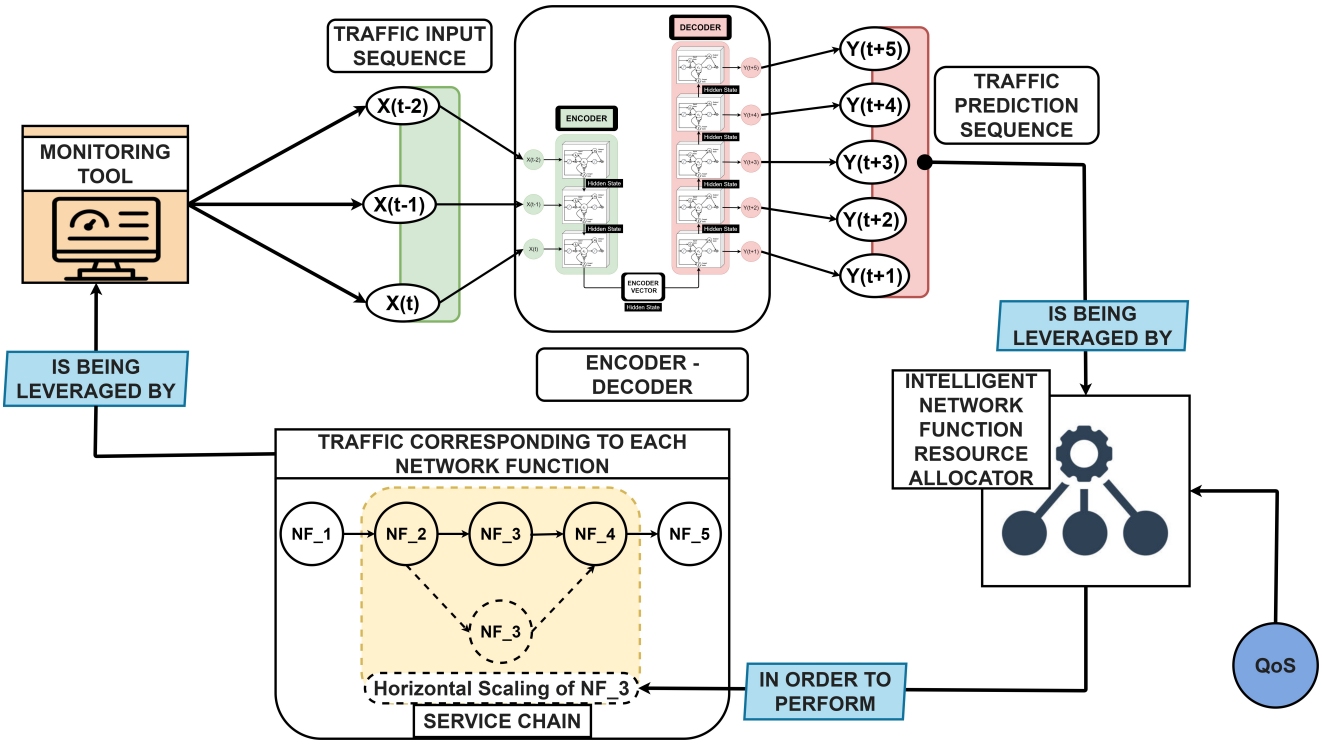


Figure 1: Leveraging Service Traffic Prediction for Horizontal Scaling of Network Functions

that this feedback approach is not directly designed for sequence prediction and as a result tends to accumulates errors over steps.

A sequence to sequence (seq2seq) architecture can capture the temporal dependencies and provide predictions for different time steps. A prominent approach for seq2seq is the encoder-decoder [15] which consists of one neural network that maps the input sequence of previous steps to an intermediate vector and the decoder which maps the intermediate vector to a sequence prediction. Encoder-decoders have been used in multiple fields for multi-step prediction but they have not been used in service traffic prediction. Especially the multi-step predictions in the domains of transportation [16] and spatio-temporal mobility [17] have many in common in regards to their perspective problem formulation and data structure with the service traffic. Encoder-decoder with a Convolution Neural Network (CNN) encoder and a RNN decoder has been used for intelligent transportation planning [18].

To summarise what motivated this research, we have seen that service traffic prediction is a key challenge in order to achieve optimal management of the resources on top of which the big data services run. The existed statistical time series solutions have been surpassed by the DL approaches. But, the DL approaches have not been extended to provide multi-step forecasting. Encoder-decoder models are a prominent approach for multi-step forecasting. To the best of our knowledge this is the first work that examines the use of encoder-decoders for multi-step service traffic prediction. In addition we examine four different types of encoder-decoder: a) Stacked LSTMS, b) CNN-LSTMS, c) Bidirectional LSTM, and d) the innovative Hybrid LSTM. Furthermore, we perform Bayesian hypertuning in order to establish near-optimal topologies for the various encoder-decoder architectures. Finally, we provide predictions for three traffic metrics: a) The number of requests, b) the amount of transmitted data, and c) the duration of the service sessions.

## 3 Problem Setup: Multi-step Traffic Prediction for Data services

We followed a univariate forecasting approach instead of combining various input features in order to ensure the generality of our method. The use of different specific input features may increase the traffic accuracy but it is not sure these features will be available in different use cases. Additionally, if the untivariate forecasting method has good results, other researchers can include more features and apply a multivariate approach. The data observations we used come from implementations of the TCP which is a highly reliable protocol built on IP for communication between application services and computing devices. TCP guarantees the integrity of data and prevents data loss, corruption, or out-of-order delivery. It also utilises packet sequence numbering and acknowledgement packets to provide successful

data delivery. Cloud and edge computing infrastructures heavily rely on the TCP/IP protocol suite [19],thus making the tracing of TCP a trustworthy source of information regarding big data services and their perspective workload.

The service traffic prediction is important for the optimal management of a) Computational, b) Storage and c) Network resources on top of which the big data services run. These resources include nodes dedicated to data processing, compression, reporting and visualization [8] and network functions such as firewalls, intrusion prevention systems, and network address translation [20]. The service traffic prediction is leveraged in order to optimize the proactive resource deployment in response to the changes in service requests and workloads by implementing a sequence of procedures of how/where to store, process and transmit data over various execution environments.

The Figure 1 illustrates a contemporary service chain scenario. The traffic monitoring tool provides the current traffic values to the encoder-decoder, which then outputs the traffic prediction sequence. The traffic prediction sequence is being leveraged by the Intelligent Network Function Resource Allocation to provide the necessary resources on the fly, thus keeping the fulfillment of QoS requirements at acceptable levels. The Intelligent Network Function Resource Allocation mechanism performs horizontal or vertical scaling, by dynamically allocating resources to keep up with the data-flows of the next time periods. Each resource has a different deployment time. For instance, a network address translation needs less than five minutes but a cloud VM requires 10-15 minutes or even more in some cases for deployment, depending on the data services it runs. Intelligent resource allocation components and predictive mechanisms can run locally in the edge processing nodes. In this research we don't examine the intelligent resource allocation mechanism but we focus on traffic prediction and the accuracy of the predictions during different time-steps.

## 4 Modelling and Forecasting Service Traffic with Time Series

The modeling of Service Traffic can be formalized as time series with the traffic variable indexed in time order. Service traffic can be analyzed with the use of exploratory and predictive methods. In exploratory analysis, we recognise repeating traffic patterns, the correlation between traffic values and deconstruct the time series into components. In predictive analysis, we predict the next-step value based on the values documented during the previous time-steps, also named lags.

The time series decomposition includes the level which represents the average value of the observations, the trend which represents the increasing or decreasing behaviour of the values, the seasonality which highlights the repeating short-term cycle in the series and the residuals which are the random variations. Service traffic is characterized by seasonality because the behaviour of service users is characterised by regular and predictable changes that recur every day or week. In a macroscopic perspective, most services have an increasing trend in regards to data and communication requirements, which manifests when examining day-to-day time series.

The main assumption in time series analysis is the stationarity. Stationarity means that the statistical properties of the observations do not change over time. Thus, time series with trends, or with seasonality, are declared not stationary. In order to model a sequence of observations as time series we should convert it to stationary by a differencing operation and then remove the trend and the seasonality aspects.

The autocorrelation depicts the relations between the current observations and the observations collected over prior time steps, while the partial autocorrelation discards the relationships of inverting observations describing the direct relationships between the observations and their lags. Positive autocorrelation can be considered as a form of persistence, namely a tendency for the data service to have the same traffic values from one time period to the next. In the experimental evaluation we will see plots of the time series decomposition and the autocorrelation.

The Auto-Regressive Moving-Average (ARMA) is a time series forecasting model consisting of the Auto-Regressive (AR) part which involves regressing the variable on its own lagged values and Moving-Average (MA) part which models the error term as a linear combination of error terms occurring contemporaneously and at various times before. The ARMA model is characterised by two values: the number of autoregressive term $p$ and the number of lagged forecast errors in the prediction equation $q$.

ARMA is used on stationary time series. If the time series is not stationary, we apply the differencing operation also named integration by taking the difference between the data values and the previous values. The number of times that the original series must be differenced in order to achieve stationarity is called the order of integration, denoted by the term $d$. The Autoregressive Integrated Moving Average (ARIMA) forecasting model is based on the ARMA in which also applied d differences to covert the time series into stationary. In order to specify the optimal ARMA and ARIMA order of the terms $p$, $q$, and $d$ there are multiple diferencing tests. For instance we can determine the order of differencing d with null hypothesis tests Kwiatkowski–Phillips–Schmidt–Shin [21], Augmented Dickey-Fuller or Phillips–Perron.

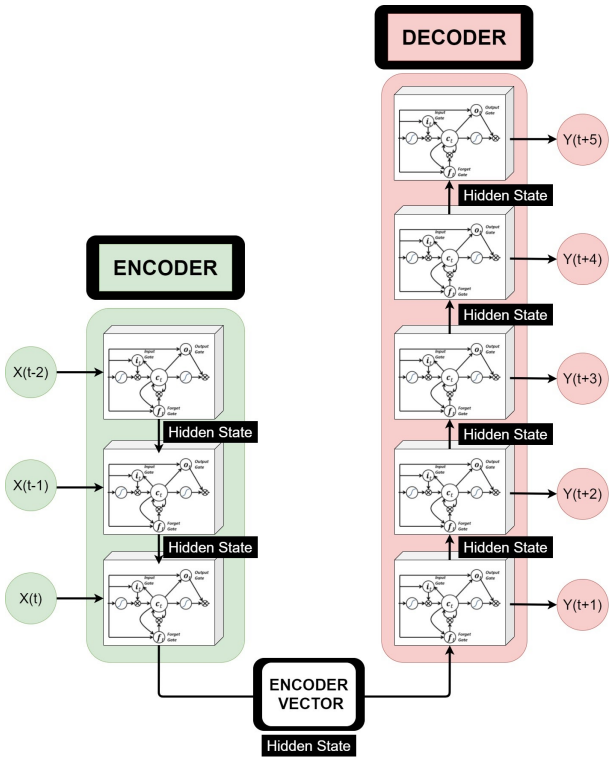


Figure 2: LSTM Encoder-Decoder

Table 1: Comparison for the first single-step prediction Statistical Time Series, Deep Learning and Encoder-Decoder methods.

| 2*Method | Requests | | Traffic | | Duration | | Training | Inference |
|---|---|---|---|---|---|---|---|---|
| | RMSE | MAE | RMSE | MAE | RMSE | MAE | Time | Time |
| ARMA | 23.149 | 16.493 | 1430557 | 937158 | 42303 | 19242 | 14.290 | 0.841 |
| ARIMA | 23.199 | 16.554 | 1424586 | 889310 | **41251** | 14026 | **13.665** | **0.297** |
| LSTM 1step | **22.344** | **15.936** | **804957** | **610643** | 42531 | 10497 | 38.458 | 2.680 |
| LSTM vec | 26.205 | 18.398 | 831477 | 645565 | 22372 | 10957 | 52.138 | 2.622 |
| ED LSTM | 26.450 | 18.977 | 836682 | 639757 | 44654 | 11390 | 136.256 | 2.684 |
| ED CNN-LSTM | 26.602 | 18.998 | 838714 | 649777 | 42499 | **10029** | 88.254 | 2.540 |
| ED Bid-LSTM | 26.734 | 19.108 | 844648 | 646605 | 42504 | 10040 | 167.308 | 2.759 |
| Hybrid | 26.052 | 18.688 | 806952 | 645182 | 42396 | 10172 | 318.303 | 2.632 |

## 5 Encoder-decoder Topologies for Multi-step Traffic Prediction

What makes encoder-decoder models an ideal candidate for sequence-to-sequence prediction is their inherit ability to map sequences of different lengths to each other. This functionality is the result of the model's architecture. The encoder takes the input sequence and represent the information as latent variables. The decoder is set to the final states of the encoder and trained to generate the output based on the information gathered by the encoder.

### 5.1 ENCODERS

#### 5.1.1 LSTM ENCODER

In order to create the encoder part of the architecture a unidirectional LSTM model was utilized. This model receives as input the values corresponding to the last 3 time-steps and produces a N element output vector which entails an internal representation of the input sequence. The size of N corresponds to the number of LSTM units used.

#### 5.1.2 Bidirectional LSTM ENCODER

Contrary to the previous one, the Bidirectional LSTM Encoder consists of a bidirectional LSTM model. This model receives as input the values corresponding to the last 3 time-steps and produces a N element output vector which entails an internal representation of the input sequence. The size of N corresponds to the number of LSTM units used.

#### 5.1.3 CNN-LSTM ENCODER

Convolution Neural Networks (CNN) are not designed to accommodate input in the form of sequences. Nevertheless a 1-dimensional CNN layer is capable of receiving as input and then learn the salient features. Additionally, both CNNs and LSTMs expect a 3-dimensional input. As far as CNNs are concerned this design characteristic is formulated in order to be able to receive the three distinct Red-Green-Blue channels. LSTMs on the other hand require a 3-dimensional input which corresponds to the a)number of samples, b) the number of time-steps to examine and c) the number of features. Two 1-dimensional convolution layers are utilized. The first one read the input sequence and projects the result onto feature maps. The second one receives as input the output of the first one and performs the same function in order to amplify any salient features. Then an max-pooling layer is used in order to accumulate features from the maps generated by the previous two layers. Finally a flatten layer is utilized to reshape the encoder output into the desired shape that can be processed by the decoder.

#### 5.1.4 DECODERS

The decoder is implemented by utilizing a LSTM model. Each unit which is a part of the decoder is expected to output a value for each one of the 5 future time-steps that are being examined. In order to do so a Repeat-Vector layer is utilized. Furthermore it is essential to incorporate two additional layers. The layers are the interpretation layer and the output layer. The purpose of the fully connected (Interpretation) layer is to interpret each time-step in the decoder output sequence and send the product to the Output layer. This specific methodology results in a single-step prediction in the output sequence. Given that the prediction of the next 5 steps is desired, it is essential to wrap both the interpretation and the output layers inside a time-distributed wrapper. By doing so, the output provided by the decoder will be processed by the same fully-connected and output layer. Thus enabling the wrapped layers to be used for each time-step by the decoder.

### 5.2 Hybrid Unidirectional-Bidirectional LSTM

This novel architectural paradigm is the product of utilizing both bidirectional and unidirectional LSTMs instead of just one of the two. The input layer is a bidirectional LSTM. A unidirectional LSTM layer is then stacked on top of the bidirectional one. The bidirectional layer will provide one hidden state output for each time-step in 3-dimensional form which is then utilized as input by the unidirectional layer. The core idea behind this architectural choice is the fact that by introducing heterogeneous layers the model will be able to exploit the temporal correlations present in the various time-series in a more sophisticated way when compared to the rest of the models. Furthermore the fact that multiple layers are being utilized allows the features of the input sequence to be represented in a more robust way. The same design logic is implemented in the decoder part in order to mirror the encoder morphology. Instead of the basic LSTM model used in the previously explored decoders, the Hybrid model utilizes a bidirectional layer stacked on top of a unidirectional layer. This structural symmetry enables the decoder to properly reconstruct the underlying temporal motifs of the input sequence.

## 6 Experimental Evaluation

The experiments took place with a real dataset with TCP traces of data services. The traffic records were aggregated into buckets of five minutes duration.

### 6.1 Evaluation Metrics

In order to evaluate the accuracy of the proposed model we used error metrics and time metric. The error metrics are the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE). The MAE expresses the average absolute difference between the target values and the predicted values. Squaring the prediction errors and averaging the squares we have the RMSE which expresses the standard deviation of the errors emphasizing on the spread out of the errors. MAE is preferred when all the errors have the same importance, while RMSE when we should penalize the large errors even if they are just a few. In our experiments, the amount of transmitted data was in terms of megabytes and the duration of the services was in seconds. In the tables II - IV, we evaluated each forecasted time step independently in

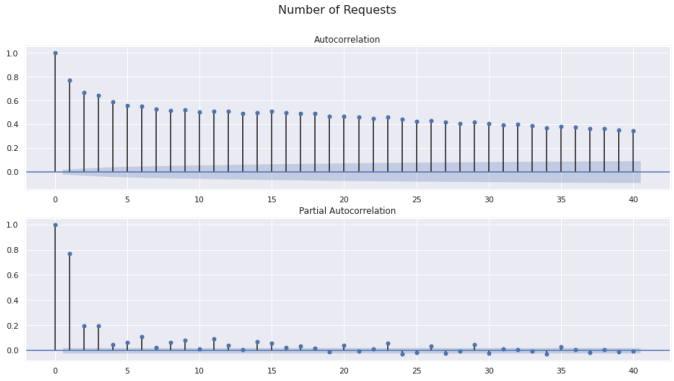


Figure 3: Number of Requests: Autocorrelation and Partial Autocorrelation

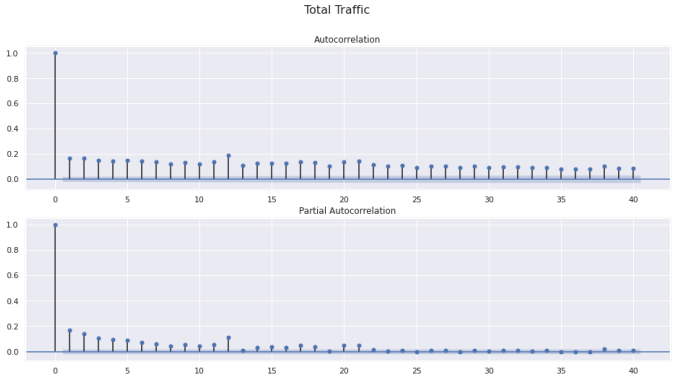


Figure 4: Total Traffic: Autocorrelation and Partial Autocorrelation

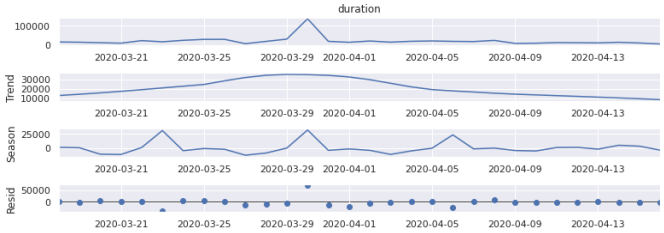


Figure 5: Duration: Decomposition in Trend, Seasonality and Residuals

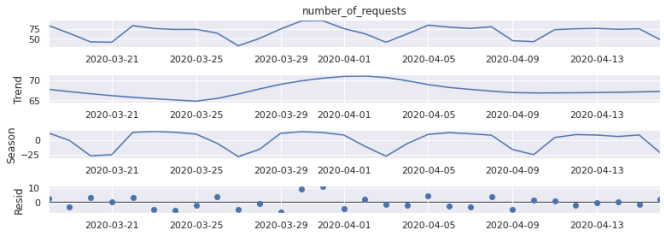


Figure 6: Number of Requests: Decomposition in Trend, Seasonality and Residuals

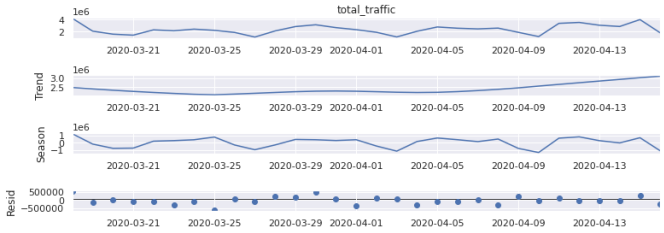


Figure 7: Total Traffic: Decomposition in Trend, Seasonality and Residuals

Table 2: Comparison of the multi-step prediction methods for the number of request.

| 2*Requests | 1st step | | 2nd step | | 3rd step | | 4th step | | 5th step | |
|---|---|---|---|---|---|---|---|---|---|---|
| | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| LSTM Vec | 22.794 | **15.704** | 25.808 | **17.885** | 26.637 | **18.956** | 27.539 | **19.378** | 28.249 | **20.066** |
| ED LSTM | 22.675 | 16.147 | 25.807 | 18.422 | 27.119 | 19.454 | 27.901 | 20.157 | 29.071 | 20.705 |
| ED CNN-LSTM | 23.348 | 16.584 | 25.877 | 18.311 | 26.906 | 19.282 | 28.006 | 20.063 | 28.873 | 20.749 |
| ED Bid-LSTM | 22.827 | 16.071 | 26.241 | 18.615 | 27.191 | 19.275 | 28.311 | 20.401 | 29.102 | 21.177 |
| ED Hybrid | **22.656** | 16.173 | **25.495** | 18.145 | **26.489** | 19.098 | **27.504** | 19.754 | **28.114** | 20.268 |

order to see the models prediction skill at each specific time-step and to contrast models based on their accuracy at different time-steps.

### 6.2 Exploratory Data Analysis

Due to the morphology of the total traffic data and the number of requests, a continuous strong correlation among the lag values occurs. This is illustrated in the Figures 3 and 4 where the x axis of the ACF and PACF plots indicates the lag at which the autocorrelation coefficient and partial autocorrelation coefficient are computed; the y axis indicates the value of the correlation (between -1 and 1). This results in having to utilize rather complex models in order to properly encapsulate the MA based on strong correlated lags derived form the autocorrelation plot, due to the high value of the MA. For this reason, the number of requests and the total traffic data are better defined by the partial autocorrelation plots which removes indirect correlations.

From the Partial Autocorrelation plot of total traffic we observe that the total traffic values form strong dependencies in the window of 2-3 lag values and then tend present deviations in their behavior. Given this observation, it is safe to conclude that in a forecasting model it is important to take into consideration the values that are up to 10-15 minutes before the current time-step. In a similar manner, the Partial Autocorrelation plot of the number of requests shows that strong partial autocorrelation relationships are formed for lag values that are equal to 1, 2 and 3. We conclude that each current value depends to a significant degree on the prior time-step, but in order to achieve greater forecasting accuracy we must include the values for the last 3 time-steps. Furthermore, by observing the plots we conclude that there is no seasonality in the data, as the distribution of lag values is random.

After examining the autocorrelation and partial autocorrelation plots there seems to be no seasonality present in regards to the data values. In order to better understand the behavioral patterns of the data over multiple time-steps, it is imperative to change the frequency of observation of periodic phenomena in order to focus initially on day-to-day level frequencies, which in theory is the level at which the various periodic phenomena regarding network traffic data tend to manifest. From the figures of the results 5, 6 and 7, periodicity appears in the total traffic and in the number of requests. Periodicity patterns have some value variations with respect to each signal period, but to the extent that we can safely

Table 3: Comparison of the multi-step prediction methods for the Traffic (Transmitted data).

| 2*Traffic | 1st step | | 2nd step | | 3rd step | | 4th step | | 5th step | |
|---|---|---|---|---|---|---|---|---|---|---|
| | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| LSTM Vec | 781968 | 605345 | 798364 | 628584 | 832546 | 652228 | 846999 | **660414** | 862397 | **669140** |
| ED LSTM | 770830 | **581789** | 801711 | **614400** | 845053 | **649910** | 876312 | 672496 | 889506 | 680190 |
| ED CNN-LSTM | 801861 | 615874 | 815760 | 631545 | 843934 | 657737 | 864654 | 671775 | 867361 | 671956 |
| ED Bid-LSTM | 785167 | 595622 | 824090 | 632736 | 855272 | 658113 | 878195 | 673668 | 880516 | 672885 |
| ED Hybrid | **757915** | 601998 | **788857** | 632948 | **812605** | 651148 | **831969** | 666148 | **843414** | 673666 |

Table 4: Comparison of the multi-step prediction methods for the Session Duration.

| 2*Duration | 1st step | | 2nd step | | 3rd step | | 4th step | | 5th step | |
|---|---|---|---|---|---|---|---|---|---|---|
| | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| LSTM Vec | 43604 | 11091 | 43085 | 10710 | 43105 | 10924 | 43105 | 10924 | 43437 | 11078 |
| ED LSTM | 44816 | 11377 | 44473 | 11354 | 44518 | 11407 | 44723 | 11398 | 44740 | 11416 |
| ED CNN-LSTM | 43232 | 10591 | 42702 | 10139 | 42409 | **10018** | **42142** | **9697** | **42012** | **9701** |
| ED Bid-LSTM | **42407** | **10125** | 42702 | 10152 | 42594 | 10125 | 42434 | 9983 | 42380 | 9814 |
| ED Hybrid | 42437 | 10179 | **42324** | **10126** | **42363** | 10212 | 42429 | 10161 | 42425 | 10184 |

infer the daily periodicity in the data values. On the contrary, when examining hour-to-hour level frequency, values form large variations, without the data converging on a constant periodic phenomenon). Finally, we observe that for day-to-day level frequency the values of the noise in the data signal be more captured in a more clear manner.

### 6.3 Forecasting Evaluation Results

Table 1 summarizes the comparison of single step forecasting. For the one-step prediction the LSTM model has better request and traffic accuracy. Making experiments with the multi-step models we saw that the encoders-decoders surpass the direct and recursive ARIMA model. So we illustrate in the tables II to IV only the encoder-decoder architectures and the LSTM Vec which makes mult-step prediction with a multi-output LSTM approach and it was the best simple RNN.

When attempting to perform multi-step prediction it is expected that metrics such as RMSE and MAE are greatly affected by how far into the future we are trying to look into. In other words, the expected RMSE and MAE of the first time-step are expected to be lower when compared to the ones of the fifth time-step. These expectations are met when examining the multi-step predictions produced by the encoder–decoder models in regards to number of requests and the transmitted data as we can see in the Table 2 and the Table 3 respectively. In regards to duration prediction across all the implemented models there seems to be a clear pattern as we can see in the Table 4. The RMSE and MAE results across the various time-steps are relatively the same. This phenomenon is caused by the fact that the duration time-series bears zero autocorrelation, thus rendering the various models unable to properly formulate predictions based on temporal patterns. Instead the models are forced to produce predictions that are in accordance with the fundamental oscillation of the duration time-series in order to minimize the loss function during the fitting phase.

The fact that the Hybrid model utilizes a greater number of layers allows it to better encapsulate the signal's characteristics when compared to the other models. This effect is amplified by the fact that the Hybrid model consists of heterogeneous layers (bidirectional and unidirectional) which allows the encapsulation of temporal motifs in a more robust manner. This claim is supported by the fact that the Hybrid model produced the best RMSE scores in regards to traffic and number of requests. On the other hand, the best MAE scores in regards to traffic and number of requests were produced by various LSTM-based models whose simpler and more shallow architecture enabled them to tune into the fundamental oscillation of the time-series. Yet the Hybrid model was able to follow the signal more accurately by being able to produce predictions closer to the actual values.

## 7 Conclusion

In this paper, we compared statistical and DL models for the multi-step traffic prediction. The encoder-decoder DL architecture surpasses other statistical methods in terms of RMSE for all the time-steps and traffic metrics. We also proposed an innovative hybrid encoder-decoder architecture with bidirectional and unidirectional LSTMs layers that in most experiments has the best accuracy. The main limitation of the proposed approach is the extra overhead of the monitoring and decision making at the edge infrastructure.

The multi-step traffic prediction can be leveraged by the contemporary edge and cloud computing infrastructures to fulfil the demanded QoS of big data services and optimize the management of the computing, storage and network resources. Our next steps is to run a simulation based on the requirements of collaborative gaming and collaborative virtual reality use cases in the CloudSimSDN simulator and evaluate the improvement in the elastic service function policies.

## Acknowledgment

This work is part of the CHARITY project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101016509".

## References


[1] Bo Tang, Zhen Chen, Gerald Hefferman, Shuyi Pei, Tao Wei, Haibo He, and Qing Yang. Incorporating Intelligence in Fog Computing for Big Data Analysis in Smart Cities. *IEEE Transactions on Industrial Informatics*, 13(5):2140–2150, October 2017. Conference Name: IEEE Transactions on Industrial Informatics.

[2] Abdulsalam Yassine, Shailendra Singh, M. Shamim Hossain, and Ghulam Muhammad. IoT big data analytics for smart homes with fog and cloud computing. *Future Generation Computer Systems*, 91:563–573, February 2019.

[3] Bala M. Balachandran and Shivika Prasad. Challenges and Benefits of Deploying Big Data Analytics in the Cloud for Business Intelligence. *Procedia Computer Science*, 112:1112–1122, January 2017.

[4] John Violos, Vinicius Monteiro de Lira, Patrizio Dazzi, Jorn Altmann, Baseem Al-Athwari, Antonia Schwichtenberg, Young-Woo Jung, Theodora Varvarigou, and Konstantinos Tserpes. User Behavior and Application Modeling in Decentralized Edge Cloud Infrastructures. In Congduc Pham, Jorn Altmann, and Jose Angel Banares, editors, *Economics of Grids, Clouds, Systems, and Services*, Lecture Notes in Computer Science, pages 193–203. Springer International Publishing, 2017.

[5] R. G. Garroppo, S. Giordano, M. Pagano, and G. Procissi. On traffic prediction for resource allocation: A Chebyshev bound based allocation scheme. *Computer Communications*, 31(16):3741–3751, October 2008.

[6] Eunsook Kim, Kyungwoon Lee, and Chuck Yoo. On the Resource Management of Kubernetes. In *2021 International Conference on Information Networking (ICOIN)*, pages 154–158, January 2021. ISSN: 1976-7684.

[7] Daniel R. Figueiredo, Benyuan Liu, Vishal Misra, and Don Towsley. On the autocorrelation structure of TCP traffic. *Computer Networks*, 40(3):339–361, October 2002.

[8] M. Adel Serhani, Hadeel T. El-Kassabi, Khaled Shuaib, Alramzana N. Navaz, Boualem Benatallah, and Amine Beheshti. Self-adapting cloud services orchestration for fulfilling intensive sensory data-driven IoT workflows. *Future Generation Computer Systems*, 108:583–597, July 2020.

[9] V. Paxson and S. Floyd. Wide area traffic: the failure of Poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995. Conference Name: IEEE/ACM Transactions on Networking.

[10] V. Eramo, T. Catena, F.G. Lavacca, and F. di Giorgio. Study and Investigation of SARIMA-based Traffic Prediction Models for the Resource Allocation in NFV networks with Elastic Optical Interconnection. In *2020 22nd International Conference on Transparent Optical Networks (ICTON)*, pages 1–4, July 2020. ISSN: 2161-2064.

[11] Prince Sekwatlakwatla, Maredi Mphahlele, and Tranos Zuva. Traffic flow prediction in cloud computing. In *2016 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, pages 123–128, November 2016.

[12] Xiaofeng Cao, Yuhua Zhong, Yun Zhou, Jiang Wang, Cheng Zhu, and Weiming Zhang. Interactive Temporal Recurrent Convolution Network for Traffic Prediction in Data Centers. *IEEE Access*, 6:5276–5289, 2018. Conference Name: IEEE Access.

[13] Filip Pilka and Miloš Oravec. Multi-step ahead prediction using neural networks. In *Proceedings ELMAR-2011*, pages 269–272, September 2011. ISSN: 1334-2630.

[14] Ali R. Abdellah, Omar Abdul Kareem Mahmood, Alexander Paramonov, and Andrey Koucheryavy. IoT traffic prediction using multi-step ahead prediction with neural network. In *2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 1–4, October 2019. ISSN: 2157-023X.

[15] Seong Hyeon Park, ByeongDo Kim, Chang Mook Kang, Chung Choo Chung, and Jun Won Choi. Sequence-to-Sequence Prediction of Vehicle Trajectory via LSTM Encoder-Decoder Architecture. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1672–1678, June 2018. ISSN: 1931-0587.

[16] Zhengchao Zhang, Meng Li, Xi Lin, Yinhai Wang, and Fang He. Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies. *Transportation Research Part C: Emerging Technologies*, 105:297–322, August 2019.

[17] Xinglei Wang, Xuefeng Guan, Jun Cao, Na Zhang, and Huayi Wu. Forecast network-wide traffic states for multiple steps ahead: A deep learning approach considering dynamic non-local spatial correlation and non-stationary temporal dependency. *Transportation Research Part C: Emerging Technologies*, 119:102763, October 2020.

[18] Zhumei Wang, Xing Su, and Zhiming Ding. Long-Term Traffic Prediction Based on LSTM Encoder-Decoder Architecture. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–11, 2020. Conference Name: IEEE Transactions on Intelligent Transportation Systems.

[19] Chunye Gong, Jie Liu, Qiang Zhang, Haitao Chen, and Zhenghu Gong. The Characteristics of Cloud Computing. In *2010 39th International Conference on Parallel Processing Workshops*, pages 275–279, September 2010. ISSN: 2332-5690.

[20] Adel Nadjaran Toosi, Jungmin Son, Qinghua Chi, and Rajkumar Buyya. ElasticSFC: Auto-scaling techniques for elastic service function chaining in network functions virtualization-based clouds. *Journal of Systems and Software*, 152:108–119, June 2019.

[21] Denis Kwiatkowski, Peter C. B. Phillips, Peter Schmidt, and Yongcheol Shin. Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, 54(1):159–178, October 1992.