



UvA-DARE (Digital Academic Repository)

Dimensioning V2N Services in 5G Networks through Forecast-based Scaling

Martín-Pérez, J.; Kondepu, K.; De Vleeschauwer, D.; Reddy, V.; Guimarães, C.; Sgambelluri, A.; Valcarenghi, L.; Papagianni, C.; Bernardos, C.J.

DOI

[10.1109/ACCESS.2022.3142346](https://doi.org/10.1109/ACCESS.2022.3142346)

Publication date

2022

Document Version

Submitted manuscript

Published in

IEEE Access

[Link to publication](#)

Citation for published version (APA):

Martín-Pérez, J., Kondepu, K., De Vleeschauwer, D., Reddy, V., Guimarães, C., Sgambelluri, A., Valcarenghi, L., Papagianni, C., & Bernardos, C. J. (2022). Dimensioning V2N Services in 5G Networks through Forecast-based Scaling. *IEEE Access*, *10*, 9587-9602. <https://doi.org/10.1109/ACCESS.2022.3142346>

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

Dimensioning of V2X Services in 5G Networks through Forecast-based Scaling

Jorge Martín-Pérez*, Koteswararao Kondepu^{†¶}, Danny De Vleeschauwer[‡], Venkatarami Reddy[§], Carlos Guimarães*, Andrea Sgambelluri[¶], Luca Valcarengi[¶], Chrysa Papagianni^{||}, Carlos J. Bernardos*

*Universidad Carlos III de Madrid, Spain, [†]Indian Institute of Technology, Dharwad, India,

[‡]Nokia Bell Labs, Antwerp, Belgium, [§]Indian Institute of Technology, Hyderabad, India,

[¶]Scuola Superiore Sant'Anna, Pisa, Italy, ^{||}University of Amsterdam, Netherlands.

Abstract—With the increasing adoption of intelligent transportation systems and the upcoming era of autonomous vehicles, vehicular services (such as, remote driving, cooperative awareness, and hazard warning) will face an ever changing and dynamic environment. Traffic flows on the roads is a critical condition for these services and, therefore, it is of paramount importance to forecast how they will evolve over time. By knowing future events (such as, traffic jams), vehicular services can be dimensioned in an on-demand fashion in order to minimize Service Level Agreements (SLAs) violations, thus reducing the chances of car accidents. This research departs from an evaluation of traditional time-series techniques with recent Machine Learning (ML)-based solutions to forecast traffic flows in the roads of Torino (Italy). Given the accuracy of the selected forecasting techniques, a forecast-based scaling algorithm is proposed and evaluated over a set of dimensioning experiments of three distinct vehicular services with strict latency requirements. Results show that the proposed scaling algorithm enables resource savings of up to a 5% at the cost of incurring in an increase of less than 0.4% of latency violations.

Index Terms—Vehicle-to-Network, Scaling, Forecasting, Time-series, Machine Learning

I. INTRODUCTION

The 5th generation (5G) of mobile communications extends the general purpose connectivity design of earlier generations to support a wide variety of use cases with a disparate set of requirements and capabilities in a single and shared physical infrastructure. Namely, enhanced Mobile BroadBand (eMBB), Ultra Reliable Low Latency Communications (URLLC) and massive Machine Type Communications (mMTC), identified as the main network services to be supported by 5G systems, have different requirements in terms of end-to-end (E2E) delay, bandwidth, availability, reliability, and device density. Such distinct services will have a significant impact on how operators manage their infrastructure, with 5G networks shifting from the monolithic architectures of previous generations to a highly modular, highly flexible and highly programmable architectures. Network Function Virtualization (NFV) and Software-Defined Networking (SDN), along with the convergence of mobile networks and Edge/Cloud infrastructures, appear as key enablers for the realization of such vision. In doing so, a custom-fit paradigm becomes available where virtual and isolated networks (the so-called network slicing

[1]) are provided over the same and shared infrastructure and tailored to particular services and their requirements.

First, services and their network slices (hereinafter referred only as services) are orchestrated in an on-demand fashion. This step can be seen as the initial dimensioning of the service and mostly relies on static and pre-defined information. And second, the service must be continuously and dynamically dimensioned to face changes in the demand, avoiding any degradation of the service performance and violation of Service Level Agreements (SLAs). To this end, more traditional scaling approaches, like static or reactive (e.g., threshold-based) solutions, have been leveraged during the service lifespan. However, these are inflexible in timely (re)allocating network resources, incapable of facing unforeseen events, especially when multiple service must coexist. Consequently, they usually result in over-scaling of the network resources and, thus, a non-optimal dimensioning, which reduces the number of services that can be supported at the same time over the same infrastructure.

An optimal dimensioning of coexistent services is then essential to maximize network resources utilization and to reduce the probability of affecting the performance of one another. Forecasting appears as a key supporting capability to aid orchestrator and management entities on their decision-making processes by estimating the future demand of running services. In doing so, these entities can take better decisions regarding the (re)allocation or scaling of resources for specific services, as they will be fed not only with the current state of the network and services but also with their forecasted (future) state. Additionally, preemptive dimensioning actions (e.g., scaling in/out or up/down) can be taken to: (i) prepare the services to the expected demand beforehand; and (ii) accommodate the time required to dimension services, which depending on the action is expected to be in the order of seconds when reconfiguring the underlying network or minutes if Virtual Network Functions (VNFs) need to be scaled or redeployed.

Vehicular-to-Network (V2N) services are one of the most significant transformations of the automobile industry, gaining increased momentum through 5G networks. V2N services are characterized by having a periodic demand variation over time and by requiring process-intensive and low-latency, reliable

computing and communication services. However, as they mostly leverage on the Edge [2] as an alternative to Cloud, the lesser available resources must be managed more efficiently. This is where this work aims to contribute, by targeting the dimensioning of V2N services through forecast-based scaling solutions. To do so, it scales the resources allocated to different VNFs, feeding the decision algorithms with road traffic forecasting information. Notwithstanding, the accuracy of the forecasting information affects the correctness of the scaling decisions. As such, several traditional time series and Machine Learning (ML)-based techniques are evaluated in terms of accuracy, ability to forecast different periods in the future, and to cope with changes in the patterns. Based on the outcomes of the former analysis, selected forecasting techniques are leveraged to improve service scaling decisions, in terms of costs savings and fulfillment of their respective SLAs, when applied to distinct V2N services.

The remainder of this paper is organized as follows: Section II goes over the related work with respect to forecasting techniques and their application for forecasting road traffic and for network/service dimensioning purposes. Section III describes the selected techniques to forecast road traffic flow, which performance is evaluated over a road traffic dataset. Later, Section IV proposes a forecast-based scaling algorithm, which makes use of the forecasting information to horizontally scale different V2N services. Finally, Section V presents the main conclusions, pointing out to future research directions.

II. BACKGROUND AND RELATED WORK

This Section overviews different approaches to implement forecasting techniques, and existing state-of-the-art on their application for road traffic forecasting and network/service dimensioning purposes.

A. Forecasting Techniques

As stated in [3], the *“every-day life presents countless situations where one must somehow estimate what will happen in the future, as a basis for reaching a decision or taking action”*. Such estimation can also be interpreted as a prediction or forecast.

Traditionally, forecasting techniques resorted to time series methods, such as Error, Trend, Seasonality (ETS), Auto Regressive Integrated Moving Average (ARIMA) [4], and Triple Exponential Smoothing (TES) (i.e., Holt-Winters) [3]. These methods usually require a limited number of computational resources and low-energy because they are mainly based on simple analytical formulas [5].

With the increasing growth of available data, forecasting can benefit from ML techniques, such as Long Short-Term Memory (LSTM) [6] and Recurrent Neural Networks (RNN). In other words, ML is empowering forecasting techniques with the means to implement complex multivariate analysis accounting for different factors that impact a specific phenomenon. However, in contrast to traditional time series techniques, ML-based forecasting requires a large number of resources and energy, especially for training. Therefore, the

available computational resources in some network segments might limit their effectiveness. A careful evaluation of the tradeoff between cost and benefits of utilizing traditional time series versus ML techniques must be conducted [7] [8], before applying them to any specific scenario.

B. Road traffic forecasting

Forecasting techniques have been widely used in road traffic scenarios, since they follow a periodic and variable pattern over time. Traditional time series were firstly address to forecast road traffic flows, with methods such as ETS, ARIMA and TES (i.e., Holt-Winters) [9] [7].

With the emergence of ML, works such as [10] and [11] appeared as the first to apply, respectively, Stacked AutoEncoders (SAEs) and Restricted Boltzmann Machine (RBM) models to forecast road traffic flows. In [12], a deep regression model with four layers (including one input, two hidden and one output layer) is used to forecast vehicle flows in a city. Other works rely on the utilization of LSTM [13] [14], Deep Belief Network (DBN) [15], Dynamic Fuzzy Neural Networks (D-FNN) [16], and Gated Recurrent Units (GRU) [17], showing promising results on the application of ML-based techniques for road traffic forecasting.

C. Forecasting, and Network and Service Dimensioning

Forecasting techniques are already used in telecommunication networks to ease and automate tasks related to the lifecycle management of networks and services. As an example, predictive analytics is a key component of the Zero touch network & Service Management (ZSM) framework envisioned by ETSI [18], as an alternative to static rule-based approaches which are inflexible and hard to manage.

In [19], Deep Artificial Neural Networks are used to forecast network traffic demands of network slices with different behaviors. Similarly, in [20], a Holt-Winters-based forecasting analyzes and forecasts traffic requests associated to a particular network slice, which is dynamically corrected based on measure deviations. Whilst the former proactively adapts the resources allocated to different services, the latter implements an admission control algorithm to maximize the acceptance ratio of network slice requests. In [21], LSTM is used by a dynamic bandwidth resource allocation algorithm, aiming to compute the best resource allocation to reduce packet drop probability.

A dynamic dimensioning of the Access and Mobility management Function (AMF), which relies on traffic load forecasting using Deep Neural Network and LSTM, is proposed in [22] and [23]. In doing so, scaling decisions can be anticipated, avoiding the increase of the attachment time of user equipment and the percentage of rejected requests. A similar solution is also proposed in [24] targeting a dynamic and proactive resource allocation to the AMF, where LSTM, CNN, and a combined CNN-LSTM are used to forecast the traffic evolution of a mobile network.

Proponents resulting from the aforementioned work already point out for the potential of forecasting techniques, either

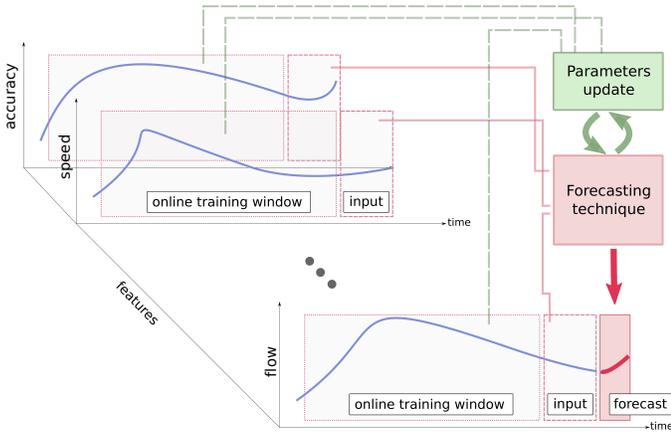


Fig. 1: Online training for traffic forecasting

traditional time series or ML-based, to provide useful information for network management operations, including the lifecycle management of services. However, to the best of our knowledge, no previous research has tackled the comparison of time series analysis and ML-based techniques, nor their performance when applied to scaling operations of V2N services with real road traffic traces. Such scenario requires traffic forecasting techniques to (i) adapt to changing road traffic conditions (as e.g. the COVID-19 lockdown witnessed in 2020), to (ii) scale vehicular services efficiently. This work addresses both challenges and, ultimately, paves the way for a scaling solution for vehicular services, namely V2N services, with strict E2E delay requirements.

III. COMPARISON OF FORECASTING TECHNIQUES

This section provides a brief description of selected forecasting techniques and how *offline/online training* can be implemented, followed by their performance using real road traffic traces.

A. Selected Forecasting Techniques

In the scope of this work, distinct time series analysis and ML-based techniques are selected, namely DES and TES time series techniques, and HTM, LSTM, GRU, TCN, and TCNLSTM ML-based techniques. These are analyzed considering two types of training: (i) an *offline training*, in which algorithms learn their parameters in the training set; and (ii) an *online training*, where the parameters are also updated as the algorithm performs forecastings (see Figure 1). To do so, the *online training* uses a moving window (called online training window – see Figure 1) comprising the most recent events, which is used to update its parameters before performing the imminent forecasting.

The next paragraphs provide an explanation of the selected forecasting techniques:

- 1) **Double Exponential Smoothing (DES) [3]:** DES is a forecasting technique based on time series analysis. DES uses a smoothing time scale with a (i) *smooth* parameter; and (ii) *trend* parameter. The smoothing time scale is

obtained based on the previous experienced time interval value of *smooth* and *trend*. In DES, the *smooth* and *trend* parameters are learned during the *offline training* stage. If DES is evaluated using *online training*, the *smooth* and *trend* values are also updated using the *online training* window for every forecast.

- 2) **Triple Exponential Smoothing (TES) [3]:** TES is another time series analysis technique. It exploits three different forecasting parameters, namely (i) *smooth*; (ii) *trend*; (iii) and *seasonality*. In TES, the *offline training* is performed by calculating *smooth*, *trend*, and *seasonality* with training set. Whereas in *online training*, the *smooth*, *trend*, and *seasonality* are updated are updated for every forecast using the online training window.
- 3) **Hierarchical Temporal Memory (HTM) [25]:** The core component of the HTM forecaster is a temporal memory consisting of a two-dimensional array of cells that can either be switched on or off and that evolves with time. Cells can influence each other via (i) *synapses* and (ii) *update rules*. The offline learning involves adjusting the *synapses* in such a way that the output bit strings resemble the actual input bit strings as much as possible. In that way the temporal memory learns to forecast the next sparse bit strings based on the patterns in the sequence of input bit string it saw. The online learning also updates the *synapses* using the *online training* window.
- 4) **Long Short-Term Memory (LSTM) [6]:** LSTM is a special form of Recurrent Neural Network (RNN) that can learn long-term dependencies based on the information remembered in previous steps of the learning process. It consists of a set of recurrent blocks (i.e., memory blocks), each of the block contains one or more memory cells, and multiplicative units with associated weights, namely, (i) *input*; (ii) *output*; and (iii) *forget gate*. LSTM is one of most successful models for forecasting long-term time series, which can be characterized by different hyper-parameters, specifically the number of hidden layers, the number of neurons, and the batch size. For the *offline training* approach neurons' weights are updated running the back-propagation-through-time [26] over a training dataset. If LSTM uses *online training*, neurons' weights use the *online training* window to update their values.
- 5) **Gated Recurrent Unit (GRU) [27]:** Gated Recurrent Units (GRUs) are neurons used in RNNs, and as LSTMs cells, they store a hidden state that is recurrently fed into the neuron upon each invocation. The neuron uses two gates, namely, (i) the *update gate*, and (ii) the *reset gate*. The former gate is an interpolator between the previous hidden state, and the candidate new hidden state; whilst the latter gate decides what to forget for the new candidate hidden state. GRUs keep track of as much information as possible of past events. Thus, their use in time-series forecasting is becoming popular in current state-of-the-art. Regarding the offline and online

training, GRU works as the aforementioned LSTM.

- 6) **Temporal Convolutional Networks (TCNs) [12]:** TCNs are deep learning architectures based on performing a temporal convolution over the input. The implemented version consists of two hidden layers, namely, (i) a first layer to perform the temporal convolution; and (ii) a second layer to re-adjust the dimension of the convolution output. In particular, the convolution layer has a window size that is a fourth of the input length in the time domain. Both the online and *offline training* update the weights of the densely connected layers, and follow the same training procedure as the LSTM solution.
- 7) **Convolutional LSTM (TCNLSTM) [28]:** In the convolutional LSTM, both TCN and LSTM models are combined into a single unified framework. The input features are fed to (i) TCN layers; and the (ii) LSTM is taking as input the output of the TCN layers to (iii) latter feed them into a dense layer. This model is considered to observe the advantage for mapping input features extraction with TCN and interpreting the features with LSTM model. In [29], it is shown that the LSTM performance can be improved by providing better features. Indeed, TCN helps by reducing the frequency variations in the input features. TCNLSTM is trained, both in the offline and online procedures, as the LSTM solution.

B. Performance Evaluation

In order to evaluate the performance of the techniques described above, a real road traffic dataset was collected from 28/01/2020 to 25/03/2020. The dataset comprises measurements from more than 100 road probes in Torino city (Italy), reporting their location, traffic flow, and vehicles speed. This dataset encompasses data pre- and post lockdown due to *COVID-19*.

Each forecasting technique is used to forecast the vehicles/hour traffic flow f_t seen at Corso Orbassano road probe¹ at time t . The set of features ϕ_i at their disposal are those reported by all road probes s_j (s_1, \dots, s_{92}) of the dataset. The numerical value of a feature reported by a probe at instant t is denoted as $x_t^{\phi_i, s_j}$. Table I enumerates the features ϕ_i , $i = \{1, \dots, 9\}$ used by the selected techniques. The dataset granularity is of 5 min., and throughout this paper $t + 1$ represents the instant $t + 5$ min.

Among all analyzed techniques, some of them can incorporate all features of past events to forecast the future flow of Corso Orbassano road. Thus, they take as input a matrix containing every feature reported by a road probe during the last h timestamps:

¹This is the road probe with highest number of reported measurements in Torino city.

TABLE I: Forecasting features

Feature	Name	Values	Description
ϕ_1	flow	integer	vehicles/hour
ϕ_2	accuracy	$\{0, \dots, 100\}$	percent accuracy of the reported measurement
ϕ_3	speed	float	average vehicles' speed (km/hour)
ϕ_4	distance to Corso Orbassano	[0,35]	distance to Corso Orbassano road probe (km)
ϕ_5	day_of_week	$\{1, \dots, 7\}$	day of the week
ϕ_6	month	$\{1, \dots, 12\}$	month of the measurement
ϕ_7	day	$\{1, \dots, 31\}$	day of the measurement
ϕ_8	year	integer	year of the measurement
ϕ_9	hour_min	[0,24)	hour+minute/60

$$X_{t,h} = \begin{pmatrix} x_{t-1}^{\phi_1, s_1} & \dots & x_{t-1}^{\phi_9, s_1} \\ \vdots & \ddots & \vdots \\ x_{t-1}^{\phi_1, s_{92}} & \dots & x_{t-1}^{\phi_9, s_{92}} \\ \vdots & \vdots & \vdots \\ x_{t-h}^{\phi_1, s_1} & \dots & x_{t-h}^{\phi_9, s_1} \\ \vdots & \ddots & \vdots \\ x_{t-h}^{\phi_1, s_{92}} & \dots & x_{t-h}^{\phi_9, s_{92}} \end{pmatrix} \quad (1)$$

Since the dataset contains periods of *COVID19* and *non-COVID19*, it is divided in two parts, each with its training and testing sets, namely:

- *non-COVID-19 scenario*:
 - training: 28th January - 28th February
 - testing: 29th February - 07th March
- *COVID-19 scenario*:
 - training: 06th February - 07th March
 - testing: 8th March - 15th March

For the performance evaluation, the *offline training* occur during the training sets to learn the techniques' weights/parameters, and the *online training* updates the learned weights/parameters as the selected technique forecasts in the testing set.

The selected techniques of Section III-A were implemented using Python and the TensorFlow library. LSTM and TCN used the whole feature matrix $X_{t,h}$ to derive the predictions, whilst the other techniques just used the traffic flow feature. Table II summarizes the parameters that allowed to get the lowest RMSE for each forecasting technique in the following experiments.

1) *Look-ahead Performance*: This Section evaluates the accuracy of the selected techniques as the look-ahead increases, i.e., as it increases the number of future traffic flow values

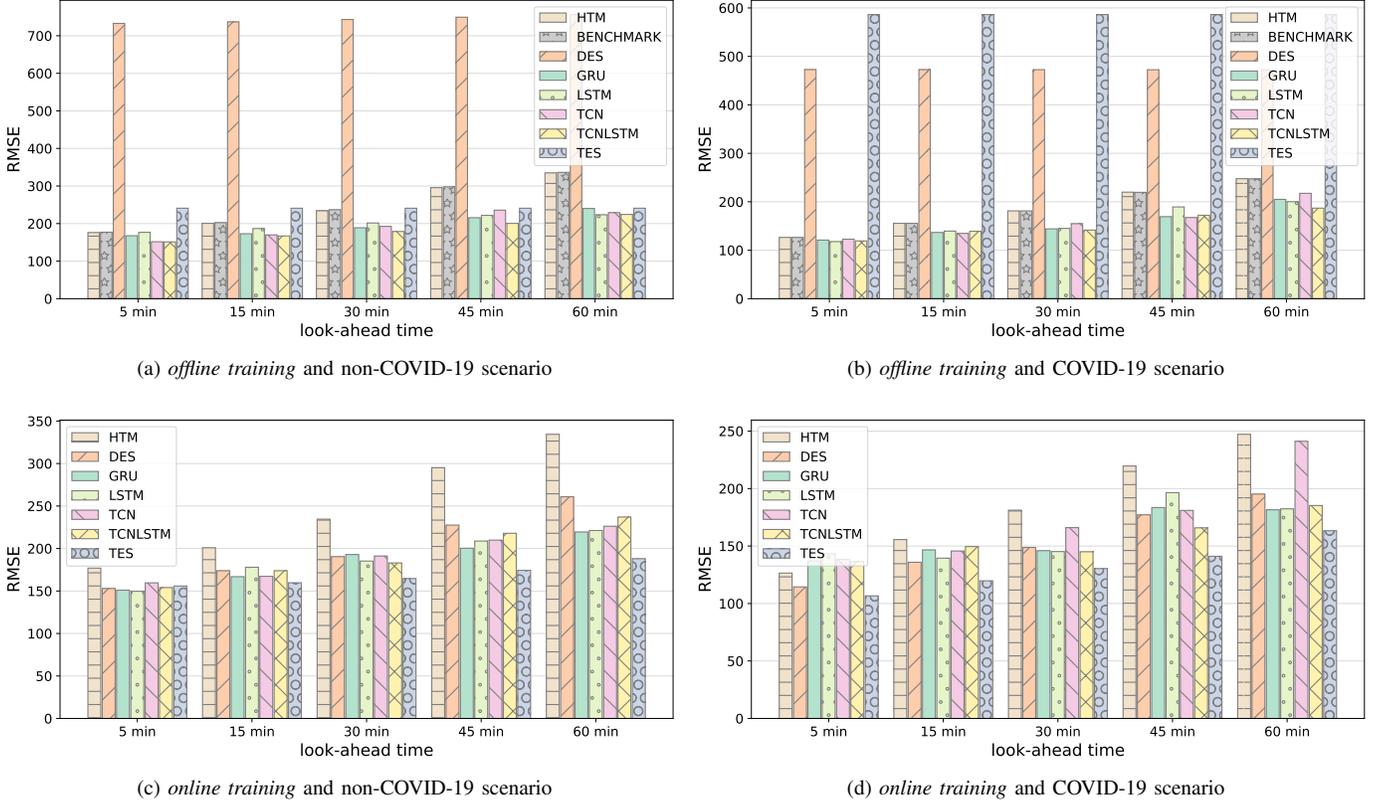


Fig. 2: Accuracy of Section III look-ahead forecasts.

TABLE II: Evaluation Parameters

Parameter	Forecasting techniques	Value
Level factor (α)	DES, TES	0.5, 0.5
Trend factor (β)	DES, TES	0.001, 0.001
Seasonality factor (γ)	TES	0.001 (3 days)
Hidden layers	TCN, LSTM, GRU, TCNLSTM	2,2, 1, 4
Neurons in hidden layer	TCN, LSTM, GRU, TCNLSTM	100
Epochs	TCN, LSTM, GRU, TCNLSTM	100
history Window size (h)	TCN, LSTM, TCNLSTM	60 min.
	GRU	120 min.
Batch size	TCN, LSTM, GRU	5
Temporal memory	HTM	32x2048
Encoder representation		1024 bit str

to predict. This analysis is of special importance given the time that it takes to (de)allocate the resources for a vehicular service given the applied virtualization technology, or the type of service. Results of Figure 2 illustrate how increasing the *look-ahead time* forecast leads to an increasing RMSE in every possible training (i.e., *online* and *offline* training) and dataset combinations (*COVID-19* and *non-COVID-19* scenario), as it

becomes more difficult to forecast the traffic further in the future.

Figures 2a and 2b show that the HTM technique did not manage to beat a sample-and-hold benchmark. Moreover, in the *online training* scenarios, it yielded the worst performance among all analyzed techniques. For the rest of the techniques, the ML-based techniques achieved the best performance for *offline training*. In the *offline training*, DES is not capable of capturing the trend, and the TES pitfalls in the *COVID-19* scenario. Unlike DES and TES, the ML-based techniques can capture the evolving traffic trend thanks to the update of their hidden states (except the TCN). This explains why the ML-based techniques achieve lower RMSE when using offline forecasting (see Figure 2a and Figure 2b). Figure 2a and Figure 2b show the RMSE values of *offline training* in *non-COVID-19* and *COVID-19* scenarios. The results presented in Figure 2a show that DES technique has highest RMSE values, because the smooth and the trend values initially calculated during the training, are not updated in the testing phase. The other time-series technique (i.e., TES) mitigates such problem since its seasonality factor can capture better the trend.

Figure 2b shows the RMSE values of the considered techniques in *offline training* with *COVID-19* traffic. The considered scenario does not show any seasonality during 8th Mar - 15th Mar due to the *COVID-19* lockdown. Thus, the obtained TES results exhibit the highest RMSE value

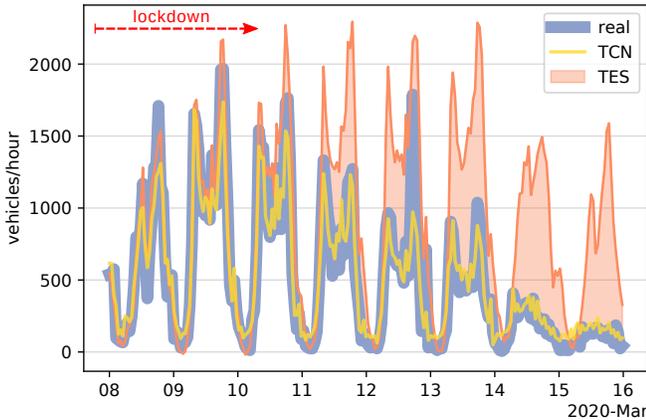


Fig. 3: TES, TCN forecasts vs. real flow values. 5 min. *look-ahead* in *COVID-19* scenario using *offline training*.

compared to all other techniques. The detailed description about this behavior is discussed later in this section.

Figure 2c and Figure 2d show the RMSE values of *online training* in *non-COVID-19* and *COVID-19* scenarios. The TES outperforms all considered ML-based techniques even when the *look-ahead time* increases. In addition, the results show that TES does not increase the RMSE as much as the ML-based techniques. This is due to the fact that it captures faster the new trends of traffic over the time. Thus, the long *look-ahead time* forecasts are better as smoothing, trend, and seasonality are updated for every data point in the test set. Even though the traditional time series techniques (i.e., DES/TES) are limited to uni-variate time series, the *online* update of their parameters achieve a better performance than the ML-based techniques that account for all features reported in Table I.

Finally, Figure 3 shows the real and the forecasted traffic flow as a function of time. Here, the *look-ahead time* is set to 5 min., and *offline training* is used to forecast the traffic flow during the *COVID-19* scenario (i.e., same conditions as in Figure 2b). It is possible to observe that the real traffic flow exhibits a seasonality pattern till 12th Mar. However, latter on traffic flow gradually decreases due to *COVID-19* lockdown. This explains why TES exhibits the highest RMSE values in Figure 2b. ML-based techniques forecast adapts to the traffic decrease, and among all them, TCN was selected to show that it forecasts traffic flow better than TES. Because every technique uses *offline training*, TES keeps using the seasonality learned during the training phase, and it forecasts high traffic flows even after the decrease.

2) *Neighborhood performance*: Results of Figure 4 show whether incorporating the information of neighboring road probes benefits Corso Orbassano traffic flow forecast. Figures 4a and Figure 4b show the impact of the neighborhood size to the RMSE using *online training* in the *non-COVID-19* scenario. The experiment considered 5 min. and 60 min. *look-ahead time* values, and quantiles in Figure 5 as neighborhood distances.

The increase of the neighborhood leads to a growth of

the training data, due to the additional information of the neighboring road probes. As shown in Figures 4a-4d, no technique is capable of reducing the RMSE by having additional neighboring information. Among all of them, DES, TES and GRU do not decrease significantly their performance. But TCN does, since it convolves every feature present in the input matrix $X_{t,h}$, including also the distance to Corso Orbassano feature. By convolving such feature over the time domain, the NN cannot distinguish whether the input corresponds to a Corso Orbassano measurement or not. This phenomenon prevents the TCN connections from giving less relevance to non-correlated measurements of irrelevant neighboring road probes. Note that HTM and LSTM results are not included, as HTM proprietary implementation could not receive multiple probes' flows as input, and both LSTM and GRU are achieving close performance results as shown in Figure 2. Most of the techniques discussed in this paper should be able to achieve the same RSME value when they take the values of more stations into account than just the Corso Orbassano station. Each technique can set the weights associated to the stations other than Corso Orbassano to 0, washing out the influence of those additional stations completely. The fact that the training does not reach this situation (where all weights associated to stations other than Corso Orbassano are set to 0) means that the training algorithm converges to local minimum rather than the global minimum, thus improvement of the training algorithm is possible.

IV. FORECAST-BASED SCALING FOR V2N SERVICES

This section devises a queuing theory-based scaling algorithm that leverages on the most accurate forecasting techniques of Section III-B.

Based on the forecasted flow of future cars, the scaling algorithm assigns enough resources to meet the vehicular service latency requirements. In particular, the performance of the proposed scaling algorithm is evaluated on three different V2N services, namely the (i) remote driving; (ii) hazard warning services; (iii) cooperative awareness, with E2E latency requirements of 5, 10 and 100 ms, respectively.

A. Forecast-based scaling algorithm

To relate number of required resources with the quality of service, a queuing model is utilized. The cars represent the clients while the available automotive service instances represent the servers. A similarity between the handover process and service request is considered. It is assumed that when cars enter in the crossing area, they are requesting the service, as if mobile users handover into another cell. For modeling the arrival process, it is necessary to select the arrival process of the cars into the service area. Previous studies have been conducted that provide careful models for automotive traffic [30]. Similarly, the service time can be modeled as the residence time in a cell of a mobile user. Previous studies provided some models as reported, for example, in [31]. In this work, the proposed model is based on the $M/M/c$ queue,

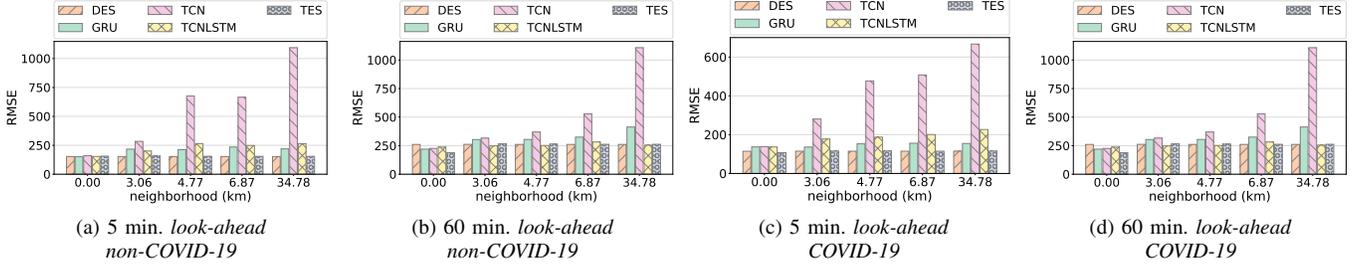


Fig. 4: Accuracy of 5 min. and 60 min. *look-ahead* forecasting from Section III, using *online training* and varying the neighboring stations.

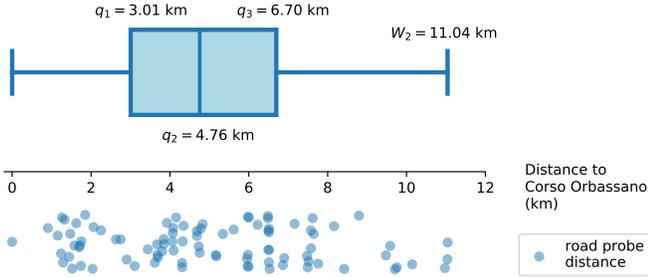


Fig. 5: Distances of dataset probes towards Corso Orbassano road probe. Boxplot above illustrates the quantiles for the distances' distribution, along with W_2 , i.e., the last road probe's distance that is below $1.5(q_3 - q_1) + q_3$.

that is at the basis of circuit switching in communications networks [32].

Thus, cars are assumed to enter in the coverage area of the service with a Poisson process with arrival rate λ and their residence time in the cell is exponentially distributed with average $\frac{1}{\mu}$. Consequently, the average time for which each vehicle spends in the system (i.e., waiting to receive the service and in the service) can be written as:

$$T = \frac{1}{\mu} + \frac{P_Q}{c\mu - \lambda}, \quad (2)$$

where c is the number of available servers and P_Q is the probability that an arrival finds all the services busy. The expression of P_Q is provided by the Erlang C formula:

$$P_Q = \frac{p_0(c\rho)^c}{c!(1-\rho)}, \quad (3)$$

where $\rho = \frac{\lambda}{c\mu}$ and the probability p_0 of having zero clients in the system is:

$$p_0 = \left[\sum_{n=0}^{c-1} \frac{(c\rho)^n}{n!} + \frac{(c\rho)^c}{c!(1-\rho)} \right]^{-1}. \quad (4)$$

Given the road traffic dataset's flow rate of vehicles $\lambda = f_t$, and the latency specification T_0 of the V2N services referred in the introduction of this section (hazard warning, cooperative

awareness, and remote driving), it is possible to derive the required number of virtualized service instances c to satisfy the average E2E latency ($T_0 = 5$ ms in the case of remote driving). More specifically, given the tuple (λ, μ) , c is increased until the average delay formula (2) reports a value of $T \leq T_0$. This is the approach used in the proposed Algorithm 1 to derive the required number of servers c and horizontally scale the V2N service. In particular, Algorithm 1 follows a n -min. horizontal scaling approach that (i) forecasts the traffic flow for the next n minutes; and (ii) scales up/down the number of servers c to meet the average delay for the maximum flow forecasted within the next n minutes.

B. Forecast-based Scaling Performance

Given the queuing theory framework, this Section studies the performance of Algorithm 1 to scale remote driving, cooperative awareness, and hazard warning V2N services.

The performance evaluation is assessed by means of cost savings, and E2E delay violations. Moreover, Algorithm 1 is compared against two other scaling strategies described latter in this Section, and during the experiments it used the most accurate forecasting technique among the ones evaluated in Section III-B. Results are derived using (i) the aforementioned road traffic dataset; and (ii) reference service rate values of a European project testbed, namely from 5G-TRANSFORMER.

In particular, the service rate μ is obtained from 5G-TRANSFORMER [33], which reports the results of what is called an Enhanced Vehicular Service (EVS), that is, a service that deploys sensing, video streaming, and processing facilities in the edge. The deliverable reports not only the required physical resources to deploy an EVS service, but as well the flow of cars used to perform their evaluations. Moreover, it details that an EVS instance, i.e. $c = 1$ in our notation, offers a service rate of $\mu_{EVS} = 208.37$ vehicles/second.

The experiments consist in running the proposed scaling Algorithm 1 in the *COVID-19* scenario. In particular, Algorithm 1 has to decide what is the required number of servers c to meet the V2N service latency constraint T_0 for the forecasted traffic flow (see Algorithm 1 line 2) within the next n minutes. The value of μ is set to be proportional to μ_{EVS} depending on the studied V2N service, and the traffic flow forecasting is done using the technique that gave the lowest

Algorithm 1: n -min. horizontal scaling

Data: n, μ, T_0

- 1 **for** $t \in \{i \cdot n/5min. : i > 0\}$ **do**
- 2 $\hat{f}_{t+n-1}, \dots, \hat{f}_{t+1} = \text{forecast}(X_{t,h});$
- 3 $\hat{F} = \max \left\{ \hat{f}_{t+k} \right\}_{k=1}^{n-1};$
- 4 $c = 1;$
- 5 $\rho = \hat{F}/c\mu;$
- 6 **while** $\frac{1}{\mu} + \frac{P_Q}{c\mu - \hat{F}} > T_0$ **do**
- 7 $c = c + 1;$
- 8 **end**
- 9 scale(c);
- 10 **end**

TABLE III: Best Traffic Flow Forecasting Techniques

Forecasting task		Best solution	
Step-ahead	Scenario	Technique	Online
5 min	non-COVID-19	LSTM	✓
	COVID-19	TES	✓
15 min	non-COVID-19	TES	✓
	COVID-19	TES	✓
30 min	non-COVID-19	TES	✓
	COVID-19	TES	✓
45 min	non-COVID-19	TES	✓
	COVID-19	TES	✓
60 min	non-COVID-19	TES	✓
	COVID-19	TCNLSTM	✓

RMSE for n minutes look-ahead predictions (see Table III). Additionally, every experiment compares the performance of the n -min. scaling Algorithm 1 with an average and maximum scaling technique. Hence, three different scaling strategies are considered for evaluation:

- *over-provisioning/max.scaling*: this strategy assumes that the V2N service is deployed with c instances capable of meeting the average E2E delay during peak hours of traffic;
- *avg. scaling*: the network dimensions the V2N service so that the c instances meet latency restrictions considering an average flow of vehicles; and
- *n -min. scaling*: using the best n -minutes ahead forecasting technique of Table III, the service is scaled to satisfy the peak of traffic forecasted for the next n minutes (see Algorithm 1).

Figure 6 compares the performance of *over-provisioning-scaling*, *avg. scaling*, and *n -min scaling* in the remote driving (see Figures 6a and 6d), cooperative awareness (see Figures 6b and 6e), and hazard warning (see Figures 6c and 6f) V2N services. The three scaling strategies were tested under simulation in the *non-COVID-19* scenario, as the traffic flow was significantly higher than in the *COVID-19*. In every simulation n was set to 30, 45, and 60 minutes for the *n -min scaling* strategy, assuming scaling operations take less than 30 minutes. Figures 6a-6c compare the cost of *avg. scaling* and *n -min scaling* against *over-provisioning scaling*; and Figures 6d-6f report the ratio of E2E violations.

For the remote driving simulations in Figures 6a and 6d, the

service rate, and target latency were set to ($\mu = \mu_{EVS}, T_0 = 5ms$). Results show that *n -min.* reduces the costs with respect to both an *over-provisioning* strategy, and the *avg. scaling*. These savings are attained during the night, when the vehicular traffic on the streets drop and it is no longer necessary to have that many computing servers c to process the traffic. Figure 7a and Figure 7b depict the service E2E delay a excess of servers using the different scaling strategies. The night saving are appreciated in the wee hours of the morning of March 3rd (see Figure 7b), when the *45 min. scaling* decreases by one the number of servers given the drop of traffic, whilst the *avg. scaling* keeps the same number of active servers, which results in a resource over provisioning leading to higher costs. Moreover, even though the *45 min. scaling* decreased the number of servers in the first hours of March 3rd, Figure 7a shows that still the service E2E delay remained below the 5 ms latency constraint. Although the reader might think that the *n -min. scaling* strategies might substantially increase the percentage of E2E delay violations, Figure 6d shows that at most, there is only an increase of $\leq 0.4\%$ of E2E delay violations over the simulated period. Additionally, such delay violations are assumable noticing the scaling savings, which go up to a 5% according to Figure 6a.

In the cooperative awareness simulations' of Figure 6b and 6e, the service rate and target latency were set to ($\mu = \mu_{EVS}/20, T_0 = 100ms$). Given the target latency $T_0 = 100ms$, throughout the experiments it was enough to deploy only $c = 1$ instances of the service almost every time. Thus, both the *average scaling*, and *over-provisioning scaling* strategies cost the same (ratio equal to 1 in Figure 6b). In the case of the *n -min. scaling* strategy, setting up n to 30 and 45 minutes result into higher costs than the *over-provisioning scaling* (ratio above 1 on the left axis in Figure 6b), as both setups over-estimate the required resources. Consequently, *n -min. scaling* leads to less E2E delay violations than *avg. scaling* (above 4 times less, $\frac{21.3108\%}{4.9045\%} = 4.34$ to be specific).

For the last V2N service, the hazard warning, every simulation used a service rate and target latency of ($\mu = \mu_{EVS}/2, T_0 = 10ms$). Figure 6f show that *n -min. scaling* with $n = 30$ and $n = 45$ achieve around 12 times ($\frac{27.5608\%}{2.2569\%} = 12.21$) less E2E delay violations than the *avg. scaling* strategy. The reduction of violations only incur in, at maximum, less than a 15 % ($\frac{0.77}{0.67} = 14.92\%$ of additional investment over the *avg. scaling* strategy. However, *n -min. scaling* underestimates the required resources when $n = 60$, and it incurs into more E2E delay violations than *avg. scaling*.

V. CONCLUSIONS AND FUTURE WORK

This paper provides an extensive analysis of state-of-the-art techniques to forecast the road traffic of Torino city, either leveraging on time-series or ML-based techniques. The performed analysis compares each forecasting technique's RMSE considering (i) forecasting intervals from 5 to 60 minutes, (ii) offline/online training; (iii) COVID-19 lockdown; and (iv) neighboring road probes. Results show that under offline

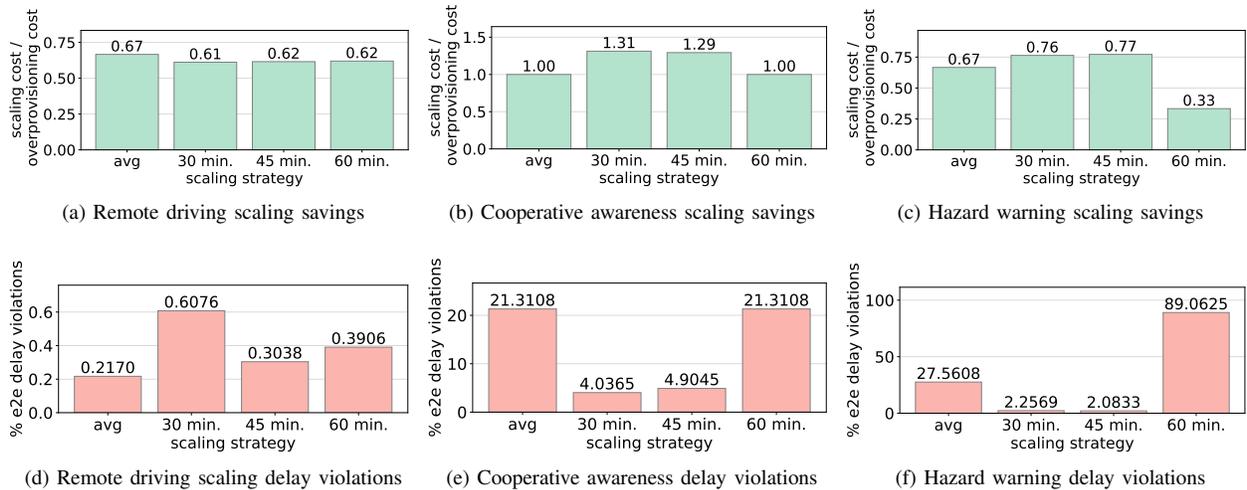


Fig. 6: Cost savings and delay violations due to scaling. TES with online training was used for n-min. strategies (i.e., Algorithm 1).

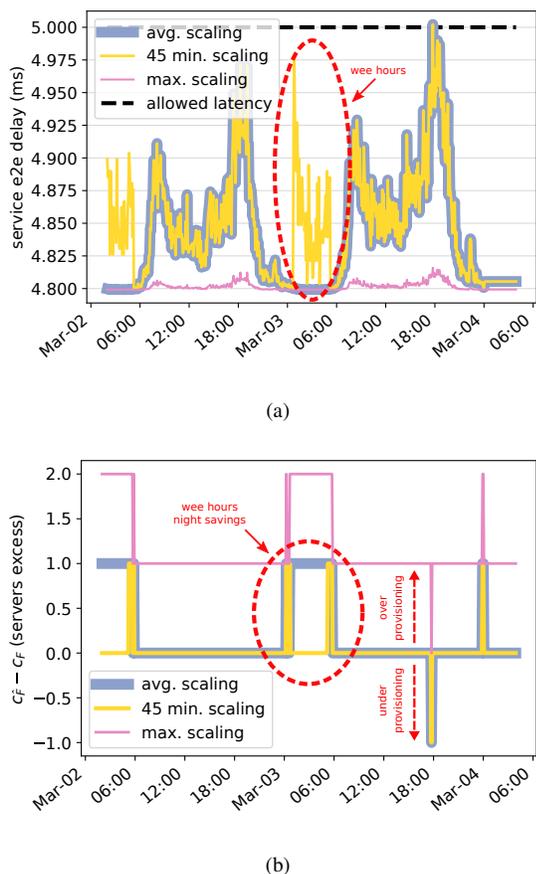


Fig. 7: Impact of remote driving scaling on (a) delay violations, and (b) difference of forecasted $c_{\hat{F}}$ and required c_F servers. TES with online training was used for 45-min. scaling (i.e., Algorithm 1).

training, ML-based techniques outperform traditional time-series methods, especially during the COVID-19 lockdown, as they adapted to the Torino traffic drop. Whilst with online training, time-series techniques achieve results better or as good as the analyzed ML-based techniques. However, none of the analyzed methods could benefit from information of neighboring stations. Experimental results confirm the benefits of applying three different forecast-based scaling solutions to dimension of V2N services. Savings of up to a 5% only incur in an increase of $\leq 0,4\%$ of latency violations in the remote driving use case. For the cooperative awareness an extra 31% of investment achieved a 4-fold latency reduction, whilst for the hazard warning less than a 15% investment increase already resulted in a 12-fold latency reduction.

A first direction to extend this work is to find techniques that can incorporate neighboring road probes' information, such as spatial analysis techniques. Furthermore, the applicability of the presented techniques to different scenarios is also envisioned as a next step of this work. The use of different datasets, including operator records with respect to the base stations used by mobile phones to access the Internet, is going to be taken into consideration. In such scenario, forecasting the user density distribution along time would enable better decisions regarding the edge server placement and service migrations.

Similarly, to the adopted scaling strategy of this work, enhancing orchestration algorithms with forecasting information would contribute to a smarter orchestration and resource control. Resulting decisions would be impacted in terms of improved quality, accuracy, and optimality. Optimized deployment, enhanced management and control of elastic network slices that support dynamic demands and their respective SLAs, improved resource arbitration and allocation and maximized service request admission are some examples where forecasting information can impact the decisions.

The aforementioned mechanisms are going to be developed and leveraged in selected use cases in the scope of the 5Growth project, which comprises Industry 4.0, transportation and energy scenarios, targeting full support of automation and SLA control for vertical services life-cycle management. Hence, it would be worth-studying the probability of forecasting less demand than what is required by each use case, i.e., $\mathbb{P}(\hat{F} < F)$; so as to perform preemptive actions under high probabilities of forecasting error. Such a calculus deserves a detailed analysis on how to compute max-statistics for correlated random variables (e.g. speed and traffic flow) [34].

ACKNOWLEDGMENT

This work has been partially funded by the EC H2020 5GROWTH Project (grant no. 856709), H2020 collaborative Europe/Taiwan research project 5G-DIVE (grant no. 859881), and “Excelencia para el Profesorado Universitario” program of Madrid regional government.

REFERENCES

- [1] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, “Network Slicing in 5G: Survey and Challenges,” *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.
- [2] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge Computing: Vision and Challenges,” *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.
- [3] R. G. Brown, *Smoothing, forecasting and prediction of discrete time series*. Prentice Hall, 1963.
- [4] Y.-S. Lee and L.-I. Tong, “Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming,” *Knowledge-Based Systems*, vol. 24, no. 1, pp. 66–72, 2011.
- [5] A. Marotta, D. Cassioli, K. Kondepu, C. Antonelli, and L. Valcarenghi, “Exploiting flexible functional split in converged software defined access networks,” *J. Opt. Commun. Netw.*, vol. 11, no. 11, pp. 536–546, 11 2019.
- [6] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, 11 1997.
- [7] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, “Statistical and machine learning forecasting methods: Concerns and ways forward,” *PLOS ONE*, vol. 13, no. 3, pp. 1–26, 3 2018.
- [8] V. R. Chintapalli, K. Kondepu, A. Sgambelluri, A. Franklin, B. R. Tamma, P. Castoldi, and L. Valcarenghi, “Orchestrating edge- and cloud-based predictive analytics services,” in *2020 European Conference on Networks and Communications (EuCNC)*, 2020, pp. 1–6.
- [9] M. Lippi, M. Bertini, and P. Frasconi, “Short-Term Traffic Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 871–882, 2013.
- [10] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, “Traffic Flow Prediction With Big Data: A Deep Learning Approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.
- [11] F. Kong, J. Li, B. Jiang, and H. Song, “Short-term traffic flow prediction in smart multimedia system for Internet of Vehicles based on deep belief network,” *Future Generation Computer Systems*, vol. 93, pp. 460 – 472, 2019.
- [12] M. Aqib, R. Mehmood, A. Albeshri, and A. Alzahrani, “Disaster management in smart cities by forecasting traffic plan using deep learning and gpus,” in *International Conference on Smart Cities, Infrastructure, Technologies and Applications*. Springer, 2017, pp. 139–154.
- [13] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, “LSTM network: a deep learning approach for short-term traffic forecast,” *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.
- [14] B. Yang, S. Sun, J. Li, X. Lin, and Y. Tian, “Traffic flow prediction using LSTM with feature enhancement,” *Neurocomputing*, vol. 332, pp. 320 – 327, 2019.
- [15] S. Goudarzi, M. N. Kama, M. H. Anisi, S. A. Soleymani, and F. Doctor, “Self-Organizing Traffic Flow Prediction with an Optimized Deep Belief Network for Internet of Vehicles,” *Sensors*, vol. 18, no. 10, 2018.
- [16] H. Li, “Research on prediction of traffic flow based on dynamic fuzzy neural networks,” *Neural Computing and Applications*, vol. 27, no. 7, pp. 1969–1980, 2016.
- [17] R. Fu, Z. Zhang, and L. Li, “Using LSTM and GRU neural network methods for traffic flow prediction,” in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, 2016, pp. 324–328.
- [18] ETSI GS ZSM 001, “Zero-touch network and Service Management (ZSM); Requirements based on documented scenarios,” V1.1.1, 2019.
- [19] D. M. Gutierrez-Estevez, M. Gramaglia, A. D. Domenico, G. Dandachi, S. Khatibi, D. Tsolkas, I. Balan, A. Garcia-Saavedra, U. Elzur, and Y. Wang, “Artificial Intelligence for Elastic Management and Orchestration of 5G Networks,” *IEEE Wireless Communications*, vol. 26, no. 5, pp. 134–141, 2019.
- [20] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs, “Mobile traffic forecasting for maximizing 5G network slicing resource utilization,” in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9.
- [21] S. Xiao and W. Chen, “Dynamic Allocation of 5G Transport Network Slice Bandwidth Based on LSTM Traffic Prediction,” in *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, 2018, pp. 735–739.
- [22] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, “Improving Traffic Forecasting for 5G Core Network Scalability: A Machine Learning Approach,” *IEEE Network*, vol. 32, no. 6, pp. 42–49, 2018.
- [23] I. Alawe, Y. Hadjadj-Aoul, A. Ksentini, P. Bertin, C. Viho, and D. Darche, “Smart Scaling of the 5G Core Network: An RNN-Based Approach,” in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.
- [24] I. Alawe, Y. Hadjadj-Aoul, A. Ksentini, P. Bertin, C. Viho, and D. Darche, “An Efficient and Lightweight Load Forecasting for Proactive Scaling in 5G Mobile Networks,” in *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2018, pp. 1–6.
- [25] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, “Unsupervised real-time anomaly detection for streaming data,” *Neurocomputing*, vol. 262, pp. 134 – 147, 2017, online Real-Time Learning Strategies for Data Streams.
- [26] Y. Chauvin and D. E. Rumelhart, *Backpropagation: theory, architectures, and applications*. Psychology press, 2013.
- [27] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” 2014.
- [28] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, “Convolutional, long short-term memory, fully connected deep neural networks,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4580–4584.
- [29] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, “How to construct deep recurrent neural networks,” in *Proceedings of the Second International Conference on Learning Representations (ICLR 2014)*, 2014.
- [30] M. Gramaglia, M. Fiore, and M. Calderon, “Measurement-based modeling of interarrivals for the simulation of highway vehicular networks,” *IEEE Communications Letters*, vol. 18, no. 12, pp. 2181–2184, 2014.
- [31] S. Kourtis and R. Tafazolli, “Modelling cell residence time of mobile terminals in cellular radio systems,” *Electronics Letters*, vol. 38, no. 1, pp. 52–54, 2002.
- [32] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks*. Prentice-Hall International New Jersey, 1992, vol. 2.
- [33] 5G-TRANSFORMER, “5G-TRANSFORMER Report on trials results,” European Commission, Tech. Rep. D5.4, 11 2019, Online available: http://5g-transformer.eu/wp-content/uploads/2019/11/D5.4-5G-TRANSFORMER_Report_on_trials_results.pdf.
- [34] S. N. Majumdar and A. Pal, “Extreme value statistics of correlated random variables,” *arXiv preprint arXiv:1406.6768*, 2014.