Multivariate Time Series Characterization and Forecasting of VoIP Traffic in Real Mobile Networks

Mario Di Mauro, *Senior Member, IEEE*, Giovanni Galatro, Fabio Postiglione, Wei Song, *Member, IEEE*, and Antonio Liotta, *Senior Member, IEEE*

***Abstract*—Predicting the behavior of real-time traffic (e.g., VoIP) in mobility scenarios could help the operators to better plan their network infrastructures and to optimize the allocation of resources. Accordingly, in this work the authors propose a forecasting analysis of crucial QoS/QoE descriptors (some of which neglected in the technical literature) of VoIP traffic in a real mobile environment. The problem is formulated in terms of a multivariate time series analysis. Such a formalization allows to discover and model the temporal relationships among various descriptors and to forecast their behaviors for future periods. Techniques such as Vector Autoregressive models and machine learning (deep-based and tree-based) approaches are employed and compared in terms of performance and time complexity, by reframing the multivariate time series problem into a supervised learning one. Moreover, a series of auxiliary analyses (stationarity, orthogonal impulse responses, etc.) are performed to discover the analytical structure of the time series and to provide deep insights about their relationships. The whole theoretical analysis has an experimental counterpart since a set of trials across a real-world LTE-Advanced environment has been performed to collect, post-process and analyze about 600,000 voice packets, organized per flow and differentiated per codec.**

***Index Terms*—VoIP traffic characterization, multivariate time series forecasting, machine learning for time series forecasting, mobility scenarios.**

I. Introduction and Motivation

PERFORMANCE prediction of real-time traffic (such as VoIP) is a crucial topic in the network management field. Predicting and optimizing Quality of Service (QoS) and Quality of Experience (QoE) metrics allows to better dimensioning network infrastructures, improving the battery life of devices, and optimizing the resource allocation strategies [1], [2]. This is even more critical in cellular environments, where the high unpredictability of variables such as the interference, but also the concurrency of real-time sessions, and the time-varying load of mobile network nodes pose intriguing challenges.

We tackle these issues through a multivariate predictive time series analysis of VoIP traffic across an urban LTE-A environment. At the moment, LTE represents the dominant broadband technology, accounting for 57% of users worldwide [3]. Older technologies such as 2G and 3G continue to be intensively used for their robustness, with about 38% of subscriptions; whereas 5G accounts for about 5% of subscriptions due to its market immaturity. Interestingly, one of the most adopted deployment today is the Non-Standalone (NSA) 5G, where a substantial part of LTE core network is reused to implement voice-based services such as VoLTE [4].

The access to LTE technology has stimulated a series of studies devoted to analyzing the performance of QoS/QoE metrics involving, for example: various deployment strategies [5], resource allocation [6], probabilistic models [7], and coexistence with other technologies [8].

On the other hand, the main contribution offered in this work pertains to a multivariate time series characterization of the dynamic (time-varying) behavior of crucial VoIP metrics which mutually influence each other. Such a cross-dependency has a great impact on forecasting, since the future values of a specific metric (e.g., the bandwidth consumption) will depend not only on the temporal evolution of the same metric, but also on the evolution of other metrics (e.g., round-trip time, jitter) for a given VoIP flow. Accordingly, we formalize analytically such a cross-dependency by means of a vector autoregressive (VAR) model, along with a set of analyses (e.g., stationarity, causality) useful to capture some insights characterizing the mutual influence among the metrics at stake. Such a formalization is then compared to classic (e.g., tree-based) and novel (e.g., deep-based) machine learning approaches.

As a first step, we carry out an experimental campaign to collect real-world mobile VoIP traffic deriving variables such as bandwidth consumption, mean opinion score (MOS) and signal-to-noise ratio (SNR), among others. In a second step, we perform a predictive analysis aimed at discovering temporal dependencies among the variables and forecast their behavior in future time periods. At this aim, we consider two

Manuscript received 24 February 2022; revised 28 March 2023; accepted 12 July 2023. Date of publication 14 July 2023; date of current version 7 February 2024. The associate editor coordinating the review of this article and approving it for publication was D. Puthal. *(Corresponding author: Mario Di Mauro.)*

Mario Di Mauro and Fabio Postiglione are with the Department of Information and Electrical Engineering and Applied Mathematics, University of Salerno, 84084 Fisciano, Italy (e-mail: mdimauro@unisa.it; fpostiglione@unisa.it).

Giovanni Galatro is with the Client Engineering Department, IBM, 20124 Milan, Italy (e-mail: galatro.giovanni@gmail.com).

Wei Song is with the College of Information Technology, Shanghai Ocean University, Shanghai 201308, China (e-mail: wsong@shou.edu.cn).

Antonio Liotta is with the Faculty of Computer Science, Free University of Bozen-Bolzano, 39100 Bolzano, Italy (e-mail: Antonio.Liotta@unibz.it).

Digital Object Identifier 10.1109/TNSM.2023.3295748

© 2023 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

approaches: *i*) a statistical approach relying on VAR models, useful to analytically describe the dependencies among interrelated time series; and *ii*) a machine learning approach, employed by turning a time series structure into a supervised learning problem. It is worth noting that a time series analysis would be of little use when dealing with data collected in controlled environments (e.g., testbeds). In such a case, in fact, the forecast would be biased since it is possible to manually tune quantities such as interference or noise figures. Conversely, in real settings we deal with uncontrollable variables, which impact the overall performance, such as: time-varying load of radio and network nodes; physical obstacles; weather conditions; and hand over procedures.

The paper is organized as follows. Section II proposes an overview of similar works, highlighting how our work contributes to the state-of-the-art. In Section III, we offer a description of the experimental environment along with details about the time series construction. In Section IV, we formulate the problem in terms of multivariate time series characterization, and we introduce statistical and machine learning (ML) based models. In Section V, we present the experimental comparison among the different forecasting techniques by taking into account both performance and times. Section VI concludes the work along with some ideas for future research.

II. Related Work and Offered Contribution

Due to the rapid evolution of telecommunication infrastructures, themes involving the network traffic characterization are becoming decisive from a network management point of view. QoS and QoE metrics, for instance, are typically used as benchmarks to evaluate the quality of a network service; thus, predicting their behavior is crucial to the aim of network optimization and protocol design. Accordingly, in this section we propose an *excursus* of relevant works in the field of traffic characterization/forecasting, where we highlight a set of novelties emerging from our work along different directions.

A first aspect concerns the network traffic forecasting through statistical models, where a common trend is to exploit autoregressive moving average (ARMA) [9], [10] or autoregressive integrated moving average (ARIMA) models [11], [12], [13]. Although based on a robust methodology, ARMA and ARIMA models allow to characterize the behavior of individual network variables (in terms of *univariate* time series models), but are not able to capture the mutual influence among the variables, which is crucial, for example, to understand the interdependency between objective indicators (e.g., bandwidth) and subjective ones (e.g., MOS).

A univariate time series perspective is adopted also by that part of the technical literature which employs machine learning models for network traffic forecasting, including neural networks [14]; support vector machines [15]; general supervised models [16]; deep learning models [17], [18], [19], [20], [21], [22], [23].

To fill this gap, we formulate the problem in terms of a *multivariate* time series, where each variable is expressed as a function of values of the variable itself and values of the other variables. This approach allows to characterize the interdependency among variables by enabling joint analyses (e.g., orthogonal impulse response analysis) which would have no meaning in a univariate setting.

Another limitation which emerges in the part of the technical literature focusing on traffic characterization (especially in mobile environments as in our case) is the lack of real-world data. This issue is typically faced through the usage of network simulators, where many variables or models are artificially generated (e.g., mobility models, interference, packet loss, data burst, weather conditions, and many others). Examples include: [24] and [25], where LTE environments are simulated through NS-2; and [26], [27], [28], where some LTE metrics are characterized via NS-3. Other works employ customized LTE simulators to model QoE [29], [30] and QoS indicators [32], [33], respectively.

Even when network experiments are carried out within real mobile scenarios [34], [35], [36], [37], a set of limited metrics are considered, often due to the fact that standard communication protocols (e.g., RTP/RTCP) allow to natively collect only classic metrics, typically relating to bandwidth consumption or network delay. To overcome such restrictions we have set up an experimental campaign where, through the RTP Control Protocol Extended Reports (RTCP-XR), we are able to analyze QoS/QoE metrics that are usually neglected in traffic characterization, including MOS, round trip delay, playout delay buffer, and SNR.

In summary, the following contributions emerge from our work. First, we formalize the multivariate time series problem of mobile VoIP traffic through the VAR model, which allows to govern analytically the forecasting process. Moreover, through specific analyses including the Dickey-Fuller test, the OLS-CUSUM test and the orthogonal impulse response, we are able to discover interesting insights and hidden relationships among the considered VoIP metrics. Then, we turn a set of machine learning techniques (random forest, recurrent networks, etc.) into forecasting methods, by reframing the multivariate time series problem into a supervised learning one through a sliding window approach. This step is needed to evaluate and compare performance and time complexity of the statistical approach against the learning-based ones. Finally, we remark that the whole time series analysis relies on an experimental campaign carried on a real-world LTE-A network. During this campaign we: *i*) collect and elaborate a series of VoIP flows exploiting different voice codecs; *ii*) elaborate a set of performance metrics (most of them neglected in classic literature) through the support of the RTCP-XR protocol.

III. Network Scenario and Time Series Construction

The location chosen for mobile VoIP traffic collection and analysis is an urban area (about 2000 people/km^2) near Salerno (Italy). Figure 1 shows the area map derived from cellmapper [38], a crowd-sourced cellular tower and coverage mapping service. The number of evolved nodes B (eNB) aimed at handling radio links amounts approximately to 100. All the VoIP traffic is collected between two nodes: a mobile node (a car traveling at about 60 km/h) and a fixed node with a station

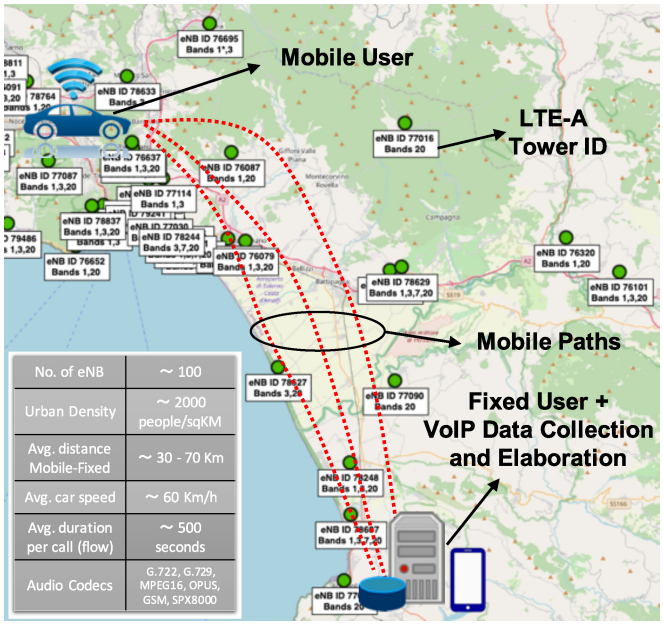


Fig. 1. Experimental setting including a mobile user (top-left) and a fixed user with a control server for data collection and elaboration (bottom-right). Main parameters are summarized in the bottom-left table.

TABLE I
SOME DATASET STATISTICS

Codec	RTP pkts	Stream length (s)	Lost pkts
G.722	55890	566	0.36%
G.729	28045	447	0.4%
MPEG-16	39181	654	0.2%
OPUS	39425	405	0.4%
GSM	38357	434	1.3%
Speex-8	43128	436	0.2%

to collect/elaborate the VoIP flows. The distance between the two nodes ranges from 30 to 70 kilometers. Both nodes are equipped with Linphone [39], one of the few softphones supporting the RTCP-XR protocol defined in the RFC 3611 [40]. Such a protocol allows to gather a rich set of metrics not available through the classic RTCP protocol, such as MOS, SNR, round trip delay (or round trip time), and playout delay buffer.

The overall collected flows amount to about 600,000 voice packets, and are divided per codecs, including the following ones: G.722 (64 kb/s of bit rate and 16 KHz of sampling rate); G.729 (8 kb/s of bit rate and 8 KHz of sampling rate); MPEG-16 (16 kb/s of bit rate and 16 KHz of sampling rate); OPUS (6 to 128 kb/s of bit rate and 48 KHz of sampling rate); GSM (8 kb/s of bit rate and 8 KHz of sampling rate); and SPX-8000 (8 kb/s of bit rate and 8 KHz of sampling rate). Such a choice is justified by the fact that each codec is able to react differently to diverse network conditions (e.g., the consumed bandwidth or the playout delay buffer) and accordingly adjust the quality of the voice flow. In this way, we obtain a more ample view of the time-based variables behavior and how these are influenced by the different codecs. In Table I we report some useful information about the collected dataset including: codec type (first column), number of RTP packets per VoIP conversation (second column), stream length, namely, the duration of conversation (third column), lost packets (fourth column). We derive such information from *RTP stream statistics* section available in Wireshark, the open source sniffer tool aimed at network traffic inspection [41]. Upon collecting the traffic, we have performed a post-processing stage to extract and process six crucial time-based variables. Precisely, for each voice flow (namely for each codec) we built a (6×1) time-based vector $y_t = (y_{1t}, \ldots, y_{6t})^T$ whose components are the following six time series:

- y_{1t}: time series representing the *MOS*, which quantifies the human subjective experience of a voice call in a dimensionless range between 1 (low perceived quality) and 5 (high perceived quality); this metric has been derived from the R-factor (R), a QoE indicator obtainable via RTCP-XR. Then, we have applied the conversion formula provided by ITU-T G.107 standard [42] to derive the MOS, namely: $\text{MOS} = 1 + 0.035R + 7{\cdot}10^{-6}{\cdot}R(R-60)(100-R)$.
- y_{2t}: time series representing the bandwidth (often *BW* for brevity) which provides information about the bandwidth consumption and is measured in kb/s;
- y_{3t}: time series representing the round-trip time (*RTT*), a key performance indicator measuring the time interval (in ms) of a voice packet sent from a source and the ack received from the destination;
- y_{4t}: time series representing the *jitter* (measured in ms), namely the variation in voice packet latency evaluated through the formula: $J_n = |(t_{r(n)} - t_{t(n)}) - (t_{r(n-1)} - t_{t(n-1)})|$ which quantifies the jitter of the n-th packet depending on the transmitting $(t_{t(n)})$ and on the receiving $(t_{r(n)})$ time;
- y_{5t}: time series representing the playout delay buffer (often *Buffer* for brevity and measured in ms), a mechanism to compensate for the encountered jitter by buffering voice packets and playing them out in a steady stream;
- y_{6t}: time series representing the signal-to-noise ratio (*SNR*), defined as the ratio between the power of a signal and the power of the background noise, and measured in decibel (dB).

Figure 2 shows all the six time series for a single voice flow (G.722 codec). Please note that this representation is meant to offer just a big picture of time series behaviors, since the measurements units are different for each series (e.g., the bandwidth is measured in kb/s, the RTT in ms, the SNR in dB, etc.). At this aim, MOS and jitter have been magnified into two separate insets (MOS in grey and jitter in light blue) to better appreciate their behaviors. We can preliminarily notice some interesting facts. For instance, it is immediate to see how the RTT time series (in green) has two noteworthy peaks (approximately at $t = 30$ and $t = 415$ seconds) probably due to some obstacles in the VoIP flow path. Yet, the bandwidth time series (in red) seems to be quite stable (around 80 kb/s) with a peak at about $t = 360$ seconds and a more irregular behavior after $t = 450$ seconds, probably due to a more unstable connection. Finally, the jitter time series is more or less regular and lying below 40 ms, as prescribed by telco standards for VoIP flows [43]. In order to inspect data variability for this flow, we also report a box-plot representation in Fig. 3. Such

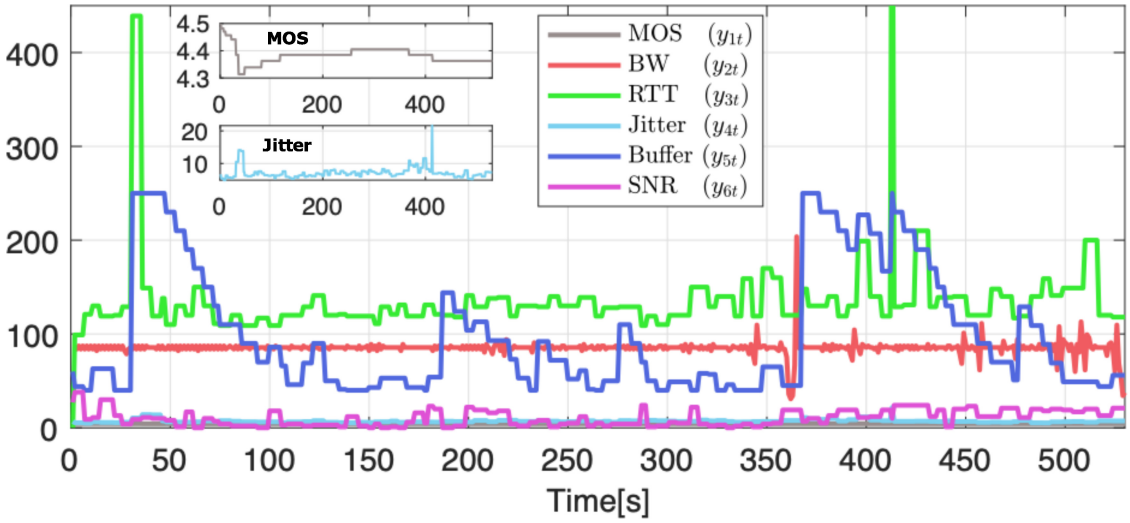


Fig. 2. Time series behavior (Codec G.722 flow) of the six crucial variables: MOS, Bandwidth, RTT, Jitter, Playout Delay Buffer, SNR.

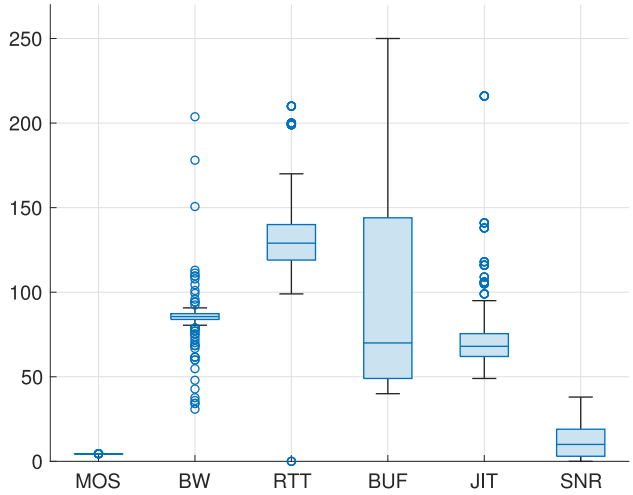


Fig. 3. Box-plot representation of time series (Codec G.722 flow).

a representation reveals that metrics such a MOS and SNR seem to be quite stable having 8 and 2 outliers (onto a stream length of 566 s, see Table I). The reason is that MOS naturally varies in a bounded range of values, whereas SNR is typically regularized thanks to the underlying codec. Remaining metrics exhibit more instability basically due to the uncontrollable external factors (e.g., interferences, mobility) thus, the number of outliers is greater: BW (39), RTT (36), Jitter (52), Buffer (97). We finally note that, to add more value to our work we make available: *i*) raw datasets divided per codec (as described in Table I); *ii*) post-processed datasets useful to directly test a given forecasting technique [44].

IV. Problem Formulation and Forecasting Models

In this section we examine in depth the problem of multivariate time series forecasting, by exploiting different techniques. Basically, through such a formalization, we try to predict future values of the time series on the basis of the current information set, namely the set consisting of current and past values of the series.

Let $y_{1t}, y_{2t}, \ldots, y_{Nt}$ be a set of N related variables. The forecast of the n-th variable $y_{n,T+H}$ at the end of period T can be expressed as:

$$\hat{y}_{n,T+H} = f\big(y_{1,T}, y_{2,T}, \ldots, y_{N,T}, y_{1,T-1}, y_{2,T-1}, \ldots, y_{N,T-1}, \ldots\big), \quad (1)$$

where H is the forecast horizon (number of future samples to predict), whereas $f(\cdot)$ is a function of the past observations and can be: *i*) a statistical model, or *ii*) derived by a machine learning algorithm. One of the most powerful classes of statistical multivariate models for time series forecasting is represented by the vector autoregressive (VAR) models. In contrast, machine learning models exploit their data-driven features to give a prediction of future values of a time series. It is useful to anticipate that to exploit a VAR model correctly, some preliminary analyses are needed (e.g., stationarity, residual correlation, etc.). Thus, in the next section we formally introduce the VAR model along with its employment into the multivariate time series field.

We should clarify that the term *variable* used in the classic statistics field is equivalent to the term *feature* typically encountered in the machine learning realm. To adopt a uniform notation, we will use the term variable (or time variable), to highlight the temporal dependency.

A. Vector Autoregressive Model

The vector autoregressive model is a generalization of the univariate autoregressive (AR) model used for time series prediction. In the classic AR model, a value from a time series is expressed in terms of its preceding values. The number of preceding values chosen for the prediction is called *order* or *lag*. We remark that, in cases where more time series are present, the AR model does not allow to capture their mutual influence. Conversely, in the VAR model a value from a time series can be expressed in terms of preceding values of the time series itself and the preceding values of other time series.

It results in a *multivariate* time series prediction problem since multiple time series can influence each other.

Let $y_t = (y_{1t}, y_{2t}, \ldots, y_{Nt})^T$ be an $(N \times 1)$ vector of time series. The p-lag vector autoregressive model VAR(p) has the following form:

$$y_t = c + \Phi_1 y_{t-1} + \Phi_2 y_{t-2} + \cdots + \Phi_p y_{t-p} + \epsilon_t, \quad (2)$$

where $c = (c_1, \ldots, c_N)^T$ denotes an $(N \times 1)$ vector of constants, Φ_k an $(N \times N)$ matrix of autoregressive coefficients $(k = 1, \ldots, p)$ estimated by estimating each equation by Ordinary Least Squares (OLS), and $\epsilon_t = (\epsilon_{1t}, \ldots, \epsilon_{Nt})^T$ an $(N \times 1)$ unobservable zero mean white noise (or residual) vector process with non singular covariance matrix $\Sigma_\epsilon = \mathbb{E}(\epsilon_t \epsilon_t^T)$, being $\mathbb{E}(\cdot)$ the expectation operator. Let $\phi_{ij}^{(k)}$ be the element of $\mathbf{\Phi}_k$ at row i and column j. For instance, a bivariate VAR(2) model can be expressed in the following matrix form:

$$\begin{bmatrix} y_{1t} \\ y_{2t} \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + \begin{bmatrix} \phi_{11}^{(1)} & \phi_{21}^{(1)} \\ \phi_{12}^{(1)} & \phi_{22}^{(1)} \end{bmatrix} \begin{bmatrix} y_{1t-1} \\ y_{2t-1} \end{bmatrix} + \begin{bmatrix} \phi_{11}^{(2)} & \phi_{21}^{(2)} \\ \phi_{12}^{(2)} & \phi_{22}^{(2)} \end{bmatrix} \begin{bmatrix} y_{1t-2} \\ y_{2t-2} \end{bmatrix} + \begin{bmatrix} \epsilon_{1t} \\ \epsilon_{2t} \end{bmatrix}. \quad (3)$$

The preliminary operation to perform when employing a VAR model is to determine the *best* lag p^*, which allows to build a VAR(p^*) model embodying most of information of the N-dimensional time series. Choosing the optimal lag length is not trivial since many criteria exist (often contradicting each other). We start by applying the Akaike Information Criterion (AIC) [45], a selection rule aimed at minimizing the forecast mean square error (MSE). Specifically, the approach is to fit VAR(p) models having orders $p = 0, \ldots, p_{max}$ and pick the value of p which minimizes the criterion.

Formally, the AIC criterion obeys to the following rule:

$$AIC(p) = \log\left|\widetilde{\Sigma}_\epsilon\right| + \frac{2pN^2}{L}, \quad (4)$$

being L the time series length and $|\widetilde{\Sigma}_\epsilon| = L^{-1}\sum_{t=1}^{L} \hat{\epsilon}_t \hat{\epsilon}_t^T$ the determinant of covariance matrix of the residuals estimated by OLS. The results of the AIC criterion applied to our VAR model made of 6 time series is shown in Fig. 4. We can see that the order p which minimizes the AIC criterion amounts to 11. Actually, when choosing the optimal lag for a VAR model, the lag resulting from a selection criteria (e.g., the AIC) represents only a preliminary choice.

Often, such a value must be adjusted to encounter also other needs [46], such as minimizing the residual correlations as explained in the next subsection.

1) Residual Analysis: When fitting a model to time series data, it is likely to find autocorrelation in the residuals (differences between observed and fitted values). In such a case, the model violates the assumption of no autocorrelation in the errors, and the forecast can be inaccurate [47]. A powerful residuals-autocorrelation test is the Breusch-Godfrey test [48] (also referred to as Lagrange Multiplier, LM test in brief), which considers the VAR model of the error vector

$$\epsilon_t = \Psi_1 \epsilon_{t-1} + \cdots + \Psi_h \epsilon_{t-h} + v_t, \quad (5)$$

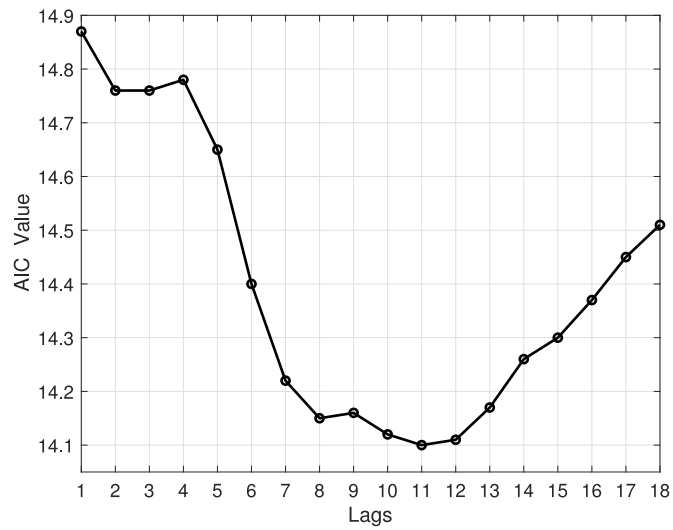


Fig. 4. AIC values for different lags applied to the considered multivariate time series model.

where h is the maximum lag of the error model and v_t a white noise at time t. In the LM test, the null hypothesis $\mathcal{H}_0$ is that there is no residual autocorrelation, whereas the alternative hypothesis $\mathcal{H}_a$ is that residual autocorrelation exists:

$$\begin{cases} \mathcal{H}_0 : \Psi_1 = \cdots = \Psi_h = 0, \\ \mathcal{H}_a : \Psi_\xi \neq 0 \quad \text{for at least one } \xi \in \{1, \ldots, h\}. \end{cases} \quad (6)$$

A common way to compute the LM test statistic based on the residuals of the VAR(p) model is to take into account the following auxiliary regression model [49]:

$$\hat{\epsilon}_t = c + \Phi_1 y_{t-1} + \cdots + \Phi_p y_{t-p} + \Psi_1 \hat{\epsilon}_{t-1} + \ldots + \Psi_h \hat{\epsilon}_{t-h} + v_t^*, \quad (7)$$

where the $\hat{\epsilon}_t$ represent the residuals from the original VAR(p) model (where $\hat{\epsilon}_t = 0$ for $t \leq 0$), and v_t^* is an auxiliary error term. Accordingly, the LM statistic can be computed as

$$Q_{LM}(h) = L\Big(N - \mathrm{tr}\Big(\tilde{\Sigma}_\epsilon^{-1} \tilde{\Sigma}_v\Big)\Big), \quad (8)$$

where $\tilde{\Sigma}_v = \frac{1}{L}\sum_{t=1}^{L} \hat{v}_t^* \hat{v}_t^{*T}$ are the residuals from the estimated auxiliary model and tr$(\cdot)$ is the trace of a matrix. Under the null hypothesis of no autocorrelation, it is possible to show [55] that $Q_{LM}(h) \xrightarrow{d} \chi^2(hN^2)$ where $\xrightarrow{d}$ indicates the convergence in distribution (as $L \to \infty$). Moreover, a correction has been proposed by Edgerton and Shukur [50] that exploits the F statistic (based on the Fisher-Snedecor distribution $F(m, l)$ with m and l degrees of freedom) in place of the χ^2, showing interesting results especially in unstable VAR models [51]. Accordingly, the Edgerton-Shukur (ES) statistic has the following form:

$$F(hN^2, \beta), \quad (9)$$

where $\beta = L - N(1 + h) + 1/2[N(h - 1) - 1]$.

Once we have chosen the optimal lag value suggested by AIC criterion $(p = 11)$, we tested the residual correlation through the hypothesis test (6). Such a test has been implemented by exploiting both χ^2 statistic of the LM test and F

TABLE II
RESULTS (IN TERMS OF P-VALUES) FOR MODELS WITH $p = 11$ AND $p = 12$ LAGS, RESPECTIVELY. TWO RESIDUAL CORRELATION TESTS HAVE BEEN CONSIDERED: LAGRANGE MULTIPLIER (LM) BASED ON THE χ^2 STATISTIC, AND EDGERTON-SHUKUR (ES) BASED ON THE F STATISTIC, BOTH COMPUTED UNDER THE NULL HYPOTHESIS

lag length=11			lag length=12		
ξ	p-value (LM)	p-value (ES)	ξ	p-value (LM)	p-value (ES)
1	0.0067	0.031	1	0.328	0.56
2	0.0044	0.036	2	0.520	0.81
3	0.0601	0.278	3	0.604	0.90
4	0.0518	0.305	4	0.642	0.94
5	0.1336	0.554	5	0.538	0.93
6	0.2292	0.733	6	0.244	0.78
7	0.1062	0.582	7	0.336	0.89
8	0.1414	0.691	8	0.399	0.93
9	0.0182	0.310	9	0.125	0.75
10	0.0097	0.255	10	0.107	0.75

statistic of the Edgerton-Shukur test. The results are shown in Table II - left side - in terms of p-values. Moreover, as suggested by credited literature [47], [52], [55], we choose a not so large value for h, namely $h = 10$.

By choosing a type I error probability $\alpha = 0.05$ to reject the null hypothesis when it is actually true, a p-value lower than α allows us to reject the null hypothesis (no residual correlation), and thus to accept the alternative hypothesis $\mathcal{H}_a$ with an error probability of 95%, at most. We highlight in red the p-values in correspondence of ξ values where the alternative hypothesis (presence of residual correlation) $\mathcal{H}_a$ of test (6) is accepted, given $\alpha = 0.05$. Such a condition occurs in both LM and ES tests, but the latter seems to be more "conservative" and allows to reject the null hypothesis less frequently than LM test. We explore also some higher lags[1] and we find interesting results for $p = 12$, which represents the second optimal choice from the AIC criterion (see Fig. 4).

The corresponding results in terms of p-values are shown in Table II - right side. It is possible to notice that the null hypothesis of no residual correlation is satisfied for all values of ξ in the case of the ES test, with p-values significantly higher than 0.05. Remarkably, also in the case of the LM test we have p-values higher than 0.05. As mentioned before, we have also tried to further increase the order p of the VAR(p) model, but we obtained more p-values allowing to reject the $\mathcal{H}_0$ hypothesis of no serial correlation (needed for accurate forecasting) than those obtained for the lag length amounting to 12, which was finally elected as the optimal choice.

2) Stationarity: When dealing with VAR models, another important operation consists in removing possible trending behaviors of the involved variables to avoid spurious regressions. Otherwise stated, we have to guarantee the *stationarity* of the time series, meaning that first and second moments must be time invariant. Pragmatically, the stationarity check is often performed through OLS-based unit root tests. In particular, Dickey and Fuller [53] developed a procedure (DF) for testing whether a variable has a unit root or, equivalently, that the variable follows a random walk.

[1]Too much high values of lags could lead the system to an overfitting.

TABLE III
AUGMENTED DICKEY-FULLER TEST PER VARIABLE

Variable	test statistic (θ)	p-value
MOS	-3.445	$9.501 \cdot 10^{-3}$
Bandwidth	-7.974	$2.718 \cdot 10^{-12}$
RTT	-2.934	$4.149 \cdot 10^{-2}$
Jitter	-4.157	$7.780 \cdot 10^{-4}$
Buffer	-2.638	$8.525 \cdot 10^{-2}$
SNR	-3.192	$2.044 \cdot 10^{-2}$

We use the augmented Dickey-Fuller test (which, differently form classic DF, includes higher-order autoregressive terms in the regression) where the following test model (in its more general form, see [54]) is considered:

$$\Delta y_t = \omega_0 + \omega_1 t + \theta y_{t-1} + \sum_{k=1}^{p} \delta_k \Delta y_{t-k} + \epsilon_t, \quad (10)$$

where: $\Delta y_t = y_t - y_{t-1}$ is the difference operator, ω_0 is the intercept term (constant), $\omega_1 t$ is the time trend, and p the lag of the autoregressive process. Finally, the test statistic on the θ coefficient is used to test whether the data need to be differenced to make it stationary. The DF test is the following:

$$\begin{cases} \mathcal{H}_0 : \theta = 0 & \text{(null hypothesis)} \quad \text{non-stationarity}, \\ \mathcal{H}_a : \theta < 0 & \text{(alternative hypothesis)} \quad \text{stationarity}. \end{cases} \quad (11)$$

For our experiments, we have performed the augmented DF test for each variable, verifying that the variables are stationary at first differences, thus, there is no need to apply the differentiation operator. The results are reported in Table III, where the negative values of θ (second column) for each variable and the corresponding low p-values (third column) suggest to reject the null hypothesis, and to accept the stationarity hypothesis by assuming a type I error probability of 0.05.

3) Stability: Stability conditions are typically required to avoid explosive solutions of the stochastic difference equations characterizing a time series expressed in terms of an autoregressive part and a moving average part. At this aim, it is possible to show [52], [55] that the VAR(p) process (2) can be written in the following $Np-$ dimensional VAR(1) form:

$$\underbrace{\begin{bmatrix} y_t \\ y_{t-1} \\ y_{t-2} \\ \vdots \\ y_{t-p+1} \end{bmatrix}}_{Y_t} = \begin{bmatrix} c \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \underbrace{\begin{bmatrix} \Phi_1 & \Phi_2 & \dots & \Phi_{p-1} & \Phi_p \\ I_N & 0 & \dots & 0 & 0 \\ 0 & I_N & \dots & 0 & 0 \\ \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & I_N & 0 \end{bmatrix}}_{\mathbf{\Phi}^*} Y_{t-1} + \begin{bmatrix} \epsilon_t \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (12)$$

being I_N the order N identity matrix. The process Y_t is stable if the eigenvalues of the *companion matrix* $\mathbf{\Phi}^*$ in (12) have modulus less than one. Such a property is satisfied for the considered VAR(p) model as can be observed in Fig. 5. Although such an analysis is formally correct to verify the stability condition, it does not allow to capture the behavior in the time

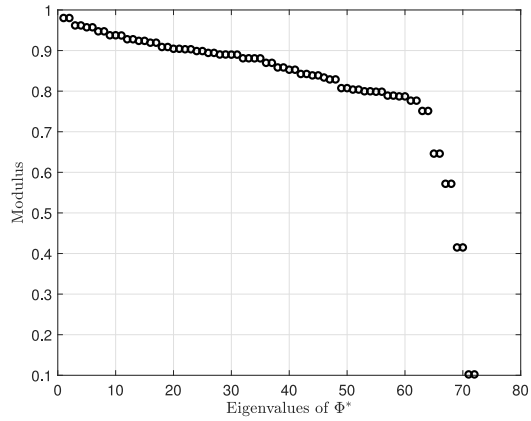


Fig. 5. Companion matrix eigenvalues.

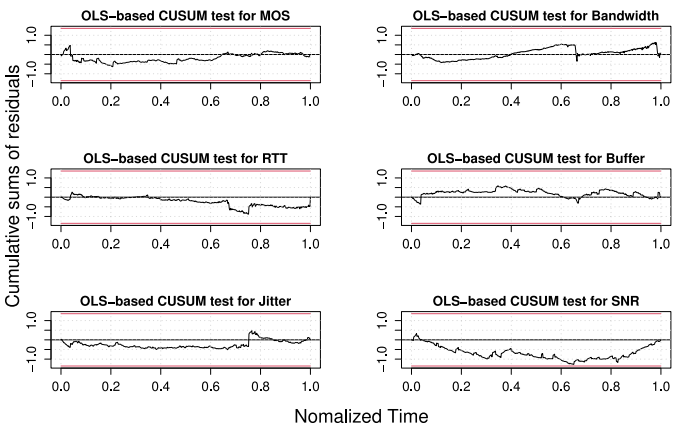


Fig. 6. OLS-CUSUM tests applied to the six time series.

domain. Accordingly, we perform in addition an OLS-based cumulative sum (CUSUM) test [56]. Through such a test, it is possible to evaluate the cumulative sums of residuals resulting from the VAR model in order to highlight potential structural changes (a.k.a. structural breaks) in the residuals which can lead to a non-stationary behavior. The test is based on the intuition that if the VAR model coefficients (the autoregressive coefficients) change over the time, the accuracy of the one-step-ahead forecast will decrease and the forecast error will increase.

The panels of Fig. 6 show the results of the OLS-based CUSUM test for all the six variables. The x-axis represents the normalized time between 0 and 1, where the y-axis reports the cumulative sums of residuals (interpretable as random processes). It is possible to notice that all the processes are substantially stable with oscillations around the zero value. A slight exception is represented by SNR where is it possible to see some little drifts from the stability value but never exceeding the 95% confidence boundaries (red lines).

4) Time Series Relationships: One of the most interesting aspects when dealing with VAR models is to understand how the time series composing the process are mutually influenced. Precisely, it is useful to know the *response* of one variable to an external disturbance (*unit impulse* or *unit shock* in the econometrics jargon) of another variable, which allows to examine more in-depth the cause/effect relation among the involved variables. In particular, if we observe a *reaction* of one variable to an impulse in another variable, the latter will be *causal* for the former [55]. In many real-world cases, there is a correlation among the variables in a system. This means that an impulse of one variable should be accompanied by an impulse of some other variables correlated with the modified one. In other words, the impulse response allows to trace the transmission of a single shock within a system of equations [57]. Often, it is interesting to isolate the effect of a single variable shock onto another variable of the system to better capture the interdependencies. At this aim, we implement the orthogonal impulse response functions (OIRF) method [58] which allows to rewrite the process (2) as:

$$y_t = c + \sum_{i=0}^{\infty} \Phi_i P P^T \epsilon_{t-i} = c + \sum_{i=0}^{\infty} \Theta_i w_{t-i}, \qquad (13)$$

where: $\Sigma_\epsilon = PP^T$ being P a lower triangular nonsingular matrix with positive diagonal elements (also known as *Choleski* decomposition, see [55, Appendix A.9.3]), $\Theta_i = \Phi_i P$ and $w_t = P^{-1}\epsilon_t$ being a white noise with covariance matrix $\Sigma_w = \mathbb{E}(w_t w_t^T) = I_K$. Since the white noise errors w_t have uncorrelated components $w_{1t}, \ldots, w_{Kt}$ with unit variance I_K, they are often known as *orthogonal* residuals or innovations. Thus, it is reasonable to assume that a change in one component of w_t has no effect on the other components due to orthogonality. In particular, the *jk*-th element of Θ_i is assumed to represent the effect on variable *j* of one unit innovation (namely, one unit standard deviation) in the *k*-th variable that has occurred *i* periods before.

In our setting, we have 6 variables resulting in 36 orthogonal impulse responses[2] as shown in the panels of Fig. 7. The causal variables are grouped per columns. The x-axis reports the observation period, thus it is possible to evaluate the disturbance effects for various observation periods (25 in our case). The blue continuous curves represent the oscillating values of the affected variables around their stability point (horizontal black line at 0), namely if the impulse were not applied at all. The black dashed curves surrounding the blue ones represent the asymptotic confidence intervals at 95% confidence level. Such an analysis has the merit of highlighting some relationships among variables which are often hidden at a first sight. For example, the sub-figure in the first row and second column allows to visualize the effect of a bandwidth shock on the MOS variable (BW $\rightarrow$ MOS). In particular, it is possible to see that a bandwidth impulse causes a slight increase of the MOS by approximately 0.006 units of innovation after about 15 observation periods. Then, it decreases. Likewise, a BW impulse causes a decrease of a couple of units of innovation in Jitter after 10 observation periods before exhausting its effect. Such behaviors are in line with the fact that having more bandwidth is typically beneficial for other metrics. It is useful to notice that, after a shock, some variables can have a decrease before raising up to their stability point. This is the case of Jitter and Buffer after a MOS impulse which experiment a decrease of 2.5 and -5 units of innovation, respectively (MOS $\rightarrow$ Jitter and MOS $\rightarrow$ Buffer sub-figures).

[2]Historically, such an analysis considers also the effect of an impulse response of a variable on itself.

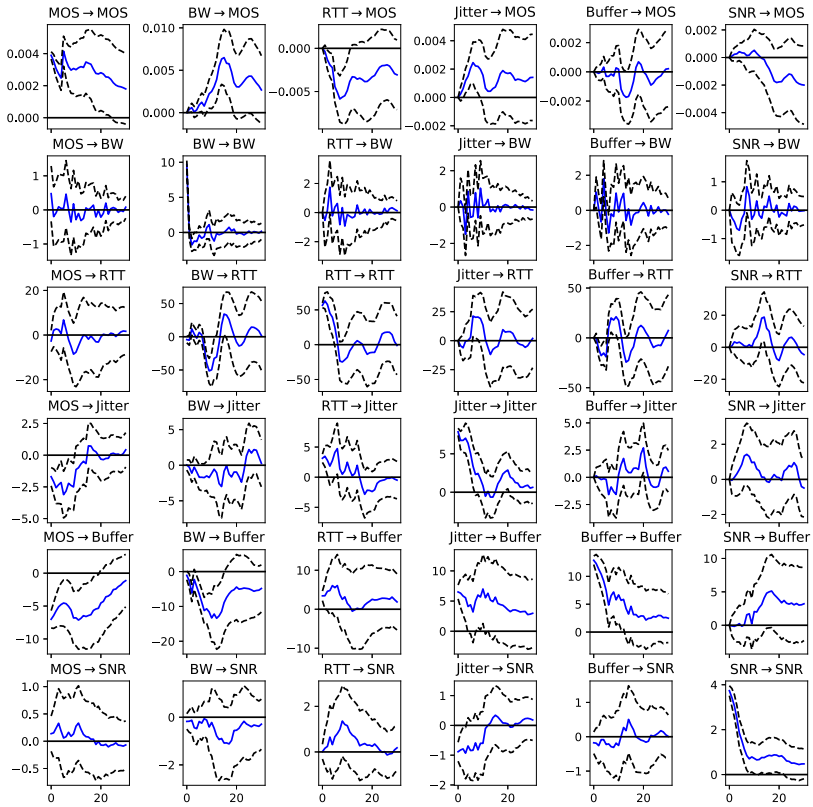


Fig. 7. Orthogonal Impulse Response.

Also in this case, we can reasonably admit that a better voice quality can be associated to lower values of jitter which, in turn, is associated to smaller values of the playout delay buffer. Interestingly, the mutual influence between two variables can be quite different when the "causing" and "caused" roles are inverted. For example, in the RTT→BW case, an RTT shock causes a slight oscillation of BW (with a peak of about 2 units of innovation around an observation period amounting to 5) before decaying rapidly to the stability point. In contrast, a BW shock causes a substantial decrease in RTT (BW→RTT) with two peaks (around −50 and 45 units of innovation) and a slower re-stabilization. Such apparently unusual behavior can be explained by the fact that BW (red curve in Fig. 2) exhibits a certain robustness, thus it is not dramatically impaired by unit shocks, whereas RTT (green curve in Fig. 2) appears to be more sensitive due to its oscillating behavior, and is then more susceptible to exogenous interventions.

B. Learning Models for Time Series Forecasting

The application of machine learning techniques to time series forecasting is a recent issue with interesting applications to econometrics [61]. When dealing with time series, in fact, the temporal information is crucial, whereas a machine learning dataset is typically a list of information equally treated from a time perspective. This notwithstanding, it is possible to manipulate these models (especially supervised ones) to train on historical time series data and provide future predictions. Moreover, some deep learning methods have been explicitly designed to take into account temporal information through memory-based cells, as detailed below.

Recurrent Neural Networks (RNNs): such a technique relies on a network architecture able to handle variable-length sequences naturally. In such a way, through the RNNs it is possible to track the state of a system (by retaining past information) and update its state as the time elapses. The memory state is recursively updated with new observations, thus the hidden state z at time t can be represented as a function of the input at time t and the previous hidden state at time $t-1$, namely:

$$z(t) = f(z(t-1), y(t)), \tag{14}$$

that, in turn, is used to evaluate the output (namely, the prediction):

$$\hat{y}_{t+1} = g(z(t)). \tag{15}$$

A weak point of RNNs is to manage long-range dependencies connected with transferring information from earlier to later times steps across too long sequences (known as the vanishing gradient problem [60]). Such an issue can be solved through the techniques explained below. The following hyperparameters have been used for RNN: 30 RNN units; dropout rate amounting to 0.25; Adam optimization algorithm (learning rate = 0.1); tanh activation function; 30 epochs.

Long Short-Term Memory (LSTM): represents an evolved RNN network [62] with some internal state cells acting as long-term or short-term memory cells. The output of the LSTM network is modulated by the state of these cells and by three *gates* which tune the information flow: the *input* gate responsible to update the cell state; the *forget* gate in charge of keeping or discarding information on the basis of the input

data $y(t)$ and the previous hidden state $z(t-1)$; the *output* gate which takes decision about which information to pass to the next LSTM unit. The hidden state at time t is:

$$z(t) = o(t) \cdot \tanh(c(t)), \tag{16}$$

being $o(t)$ the output gate, and $c(t)$ the cell state at time t. The following hyper-parameters have been used for LSTM: 30 LSTM units; dropout rate amounting to 0.25; Adam optimization algorithm (learning rate = 0.1); tanh activation function; 30 epochs.

Gated Recurrent Unit (GRU): a lighter version of LSTM [63] with two gates. The *update* gate which embodies functionalities offered by the LSTM forget and input gates. The *reset* gate which is in charge of deciding how much past information to forget. The GRU hidden state can expressed as:

$$z(t) = (1 - u(t))z(t-1) + u(t)\widetilde{z}(t), \tag{17}$$

where $u(t)$ is the update gate which decides about the updating amount of its candidate activation $\widetilde{z}(t)$. The following hyper-parameters have been used for GRU: 30 GRU units; dropout rate amounting to 0.25; Adam optimization algorithm (learning rate = 0.1); tanh activation function; 30 epochs.

Convolutional Neural Networks (CNN): they are typically used when dealing with classification problems involving spatial information where an image matrix (2D array) is provided to the CNN structure. On the other hand, when dealing with time-series problems, CNNs can be feed with a 1D array since only temporal dimension must be taken into account. Also when applied to temporal data, the CNN uses: *i*) the convolutional layer aimed at applying filtering to derive the most representative features; *ii*) the pooling layer to reduce the size of the series while preserving important extracted from convolutional layers; *iii*) the fully connected layer to map the features extracted by the network into specific classes or values. The following hyper-parameters have been used for CNN: 30 CNN filters (each of which with size 6); dropout rate amounting to 0.25; Adam optimization algorithm (learning rate = 0.1); tanh activation function; 30 epochs.

Multi Layer Perceptron (MLP): it is the most common form of neural networks and one of the first to be exploited in time series forecasting problems. The lagged observations (say x_i) are used as inputs of an MLP structure to evaluate the forecast $\hat{y}_{t+1}$:

$$\hat{y}_{t+1} = \phi\left(\sum_{i=1}^{n} w_i x_i + b\right), \tag{18}$$

where $\phi(\cdot)$ is an activation function (e.g., sigmoid, linear, etc.) to produce the output, and w_i and b are the weights and bias, respectively. The input data activate the hidden layers (intermediate layers) by following the forward activation direction, and, in turn, hidden layers neurons feed forward into output neurons. The MLP process is regulated by the *backpropagation*, a mechanism able to update neurons weights to progressively minimize the error. The following hyper-parameters have been used for MLP: 30 dense units; dropout rate amounting to 0.25; Adam optimization algorithm (learning rate = 0.1); tanh activation function; 30 epochs.

Random Forest (RF): a technique based on the *bootstrap aggregation* over decision trees. In practice, during the training stage, each *tree* within a random forest learns from random samples drawn with replacement (*bootstrapping*) so as to reach a lower variance. For each sample b, $b = 1, \ldots, B$, the desired forecast is the average of forecasts of each tree applied to the input data x_i, namely

$$\hat{y}_{t+1} = \frac{1}{B}\sum_{b=1}^{B} \hat{f}_b(x). \tag{19}$$

The following hyper-parameters have been used for RF: 30 estimators (or trees); 10 as the maximum depth of the tree.

Extreme Gradient Boosting (XGB): an improved version of gradient boosting, an iterative technique allowing to fit a decision tree on the residuals obtained from a base learner aimed at improving the prediction model by adding new decision trees. The output forecast can be written as:

$$\hat{y}_{t+1} = \sum_{k=1}^{K} \hat{f}_k(x), \tag{20}$$

where K is the number of trees and f_k is the base learner. An objective function is used to train the model by measuring how well it fits the training data. The following main hyper-parameter has been used for XGB in our experiment: 30 gradient boosted trees.

V. Experimental Forecasting Results

In this section we present a comparative analysis of the methods described in the previous section based on the experimental measurements. Before delving into details of the numerical results, we need to provide some clarifications about the processing we have performed on the gathered data.

A preliminary operation is to re-frame the time series forecasting into a supervised learning problem. We first split the multivariate time series into training and testing sets, by adopting the classic 70/30 split (70% of data is used for training, 30% for testing) as shown in Fig. 8. It is useful to highlight that the classic k-fold cross validation method cannot be applied in this setting, which assumes that there is no relationship among the observations. In contrast, when dealing with time series problems, the temporal *continuum* has to be preserved. Accordingly, we adopt the sliding window mechanism where a part of the input sequence (window of lagged values represented by the past observations within shaded blue area in Fig. 8) is used to forecast new samples (future observations within shaded red area in Fig. 8). The sliding window approach has been profitably employed also in other fields involving time series forecasting such as the smart manufacturing [64] or radar [65]. Moreover, as in the aforementioned works, we perform the so-called one-step forecasting where the prediction is made one step at a time to avoid the forecast uncertainty [47].

As regards the tuning of the various learning-based techniques, we have empirically chosen their structures so that the resulting accuracy would be in the same range of the VAR model (which does not require any fine tuning other than the

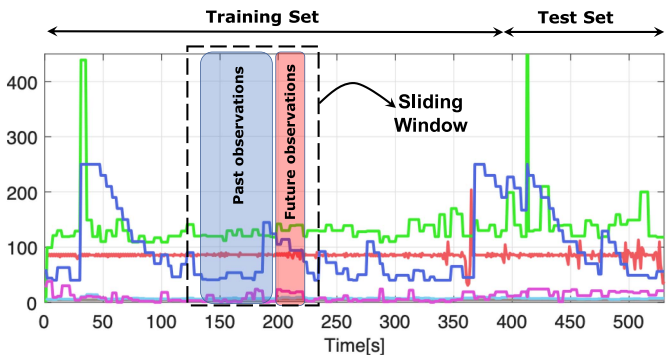


Fig. 8. Time series reframed into supervised learning through the sliding window method.

choice of the optimal lag p^*). Such an approach is in line to what suggested by credited literature [66].

We start by visually analyzing the behavior of the various presented techniques for two specific VoIP flows identified by their specific codecs, namely G.722 and G.729 (for space constraints we omit the visualization for remaining codecs but a summary of performance results for each codec is reported in the Table IV). The two aforementioned codecs represent two extreme trade-off choices between conversation quality and bandwidth utilization. Indeed, among the codecs used in our experiments, G.722 provides the better audio quality (for instance, in terms of MOS) but the bandwidth consumption is not very efficient (bit rate of 64 kb/s). In contrast, G.729 offers a slightly lower audio quality but allows a greater bandwidth saving with just 8 kb/s of bit rate.

The panels of Figs. 9 and 10 show the temporal behavior of each variable for codecs G.722 and G.729, respectively. Superimposed onto the actual values of variables (black dashed lines) we report, with different colors as specified in the figures legends, the behavior of each forecasting technique described in the previous section. In each panel of Figs. 9 and 10, the forecasting zone (the gray area on the right) defines the area where each technique tries to predict future values.

In order to highlight the behavior of VAR compared to the learning techniques, we also report, in shaded pale red, the 95% forecast intervals (for the VAR) which represent an estimate of the intervals where we expect a future value will fall. Since the interval width amounts to $1.96 \cdot \sigma_\epsilon$ (with σ_ϵ the standard deviation of residuals for each time series, see [55]), the shape and the width of each interval strongly depends on the residuals behavior. For instance, as regards the MOS - G.722 codec case (see the first panel in Fig. 9), the low residual standard deviation directly results in a narrow forecast interval. Conversely, the unexpected peak of RTT - G.722 codec case (see the third panel in Fig. 9) at about 415 seconds has a negative impact on the prediction accuracy (for all examined techniques) and, in turn, implies a growth of residual standard deviation. This directly translates into a broad forecast interval.

The first aspect to highlight is that the actual behavior of variables basically depends on two factors which impact on the performance forecast: the codec type and the network conditions. For instance, the overall bandwidth consumption amounts, on average, to 90 kb/s in the G.722 case (top-middle panel of Fig. 9), whereas it is around 30 kb/s in the G.729 case (top-middle panel of Fig. 10). In principle, this implies that more fluctuations are possible in the G.722 case due to a wider span of values. Unfortunately, even if a codec is able to guarantee a kind of temporal "stability", the high unpredictability of network conditions is the main responsible of fluctuations which are very challenging to predict due to their extremely time-variant behavior.

This notwithstanding, in both Figs. 9 and 10 we can observe that each technique is able to produce a satisfying forecast of the original variables by remaining within the area delimited by the forecast intervals.

By visual inspection, we observe that the VAR (red curve) shows good performance when the time series do not exhibit excessive fluctuations. This notwithstanding, when important fluctuations are present, the VAR model is able to follow the mean value of the oscillating time series (see, for instance, the case of MOS - G.729 codec case). The reason is that VAR is governed by a set of linear equations (see (2)), thus it can suffer when representing some non-linear behaviors. Occasionally, also the MLP technique shows slight difficulty to fit the original values, once again due to the underlying linear model (see (18)). Such a behavior emerges in particular in Fig. 9 where the MLP prediction moves a bit away from the corresponding forecast intervals. Conversely, remaining techniques show enough good adaptation to fluctuations, and, in particular, the deep-based techniques whose internal structure allows to keep the state at time $t-1$ to improve the prediction at time t.

To better quantify the behavior of each technique, we have evaluated the performance for each voice flow (namely for each codec), for each technique, and for each time-based variable as shown in Table IV. Each sub-table contains the performance per voice flow in terms of test Root Mean Square Error (RMSE), test Mean Absolute value of Errors (MAE), Mean Absolute Percentage Error, defined, respectively, as

$$RMSE_j = \sqrt{\frac{\sum_{t=1}^{L}\left(y_{jt} - \hat{y}_{jt}\right)^2}{L}}, \tag{21}$$

$$MAE_j = \frac{\sum_{t=1}^{L}\left|y_{jt} - \hat{y}_{jt}\right|}{L}, \tag{22}$$

$$MAPE_j = \frac{100}{L}\sum_{t=1}^{L}\left|\frac{y_{jt} - \hat{y}_{jt}}{y_{jt}}\right|. \tag{23}$$

Such metrics are computed for each time series $j = 1, \ldots, N$, with $N = 6$ and L the time series length. These three indicators are often used jointly when evaluating the forecasting accuracy. The RMSE is a quadratic score rule which gives a relatively high weight to large errors since these errors are squared before they are averaged. The MAE is a linear score rule designed to equally weight the individual differences. The MAPE includes a normalization to actual values and is expressed as percentage. Being such an indicator often used as a summarizing metric, to easily pinpoint the best

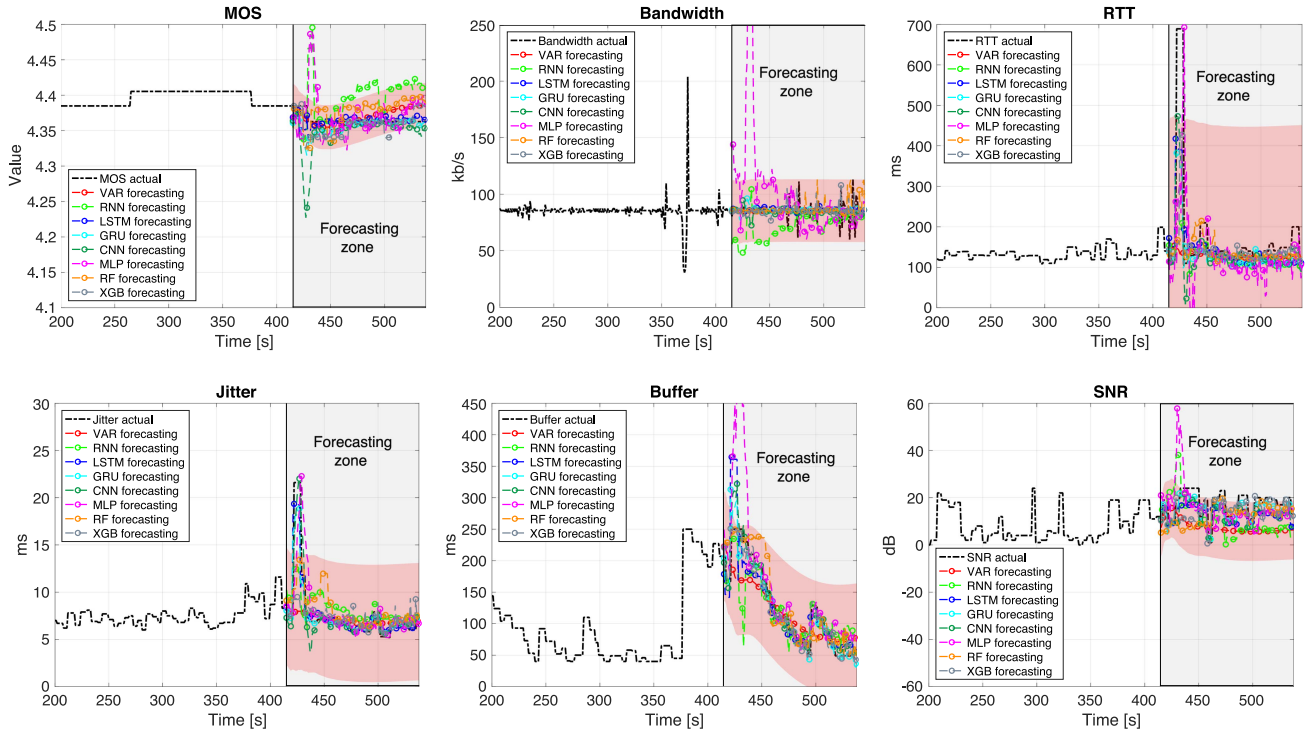


Fig. 9. G.722 codec: Multivariate time series forecasting for: MOS, Bandwidth, RTT, Jitter, Buffer, SNR, along with the 95% forecasting intervals in shaded pale red. In the gray area on the right, the results of forecasting for each technique.

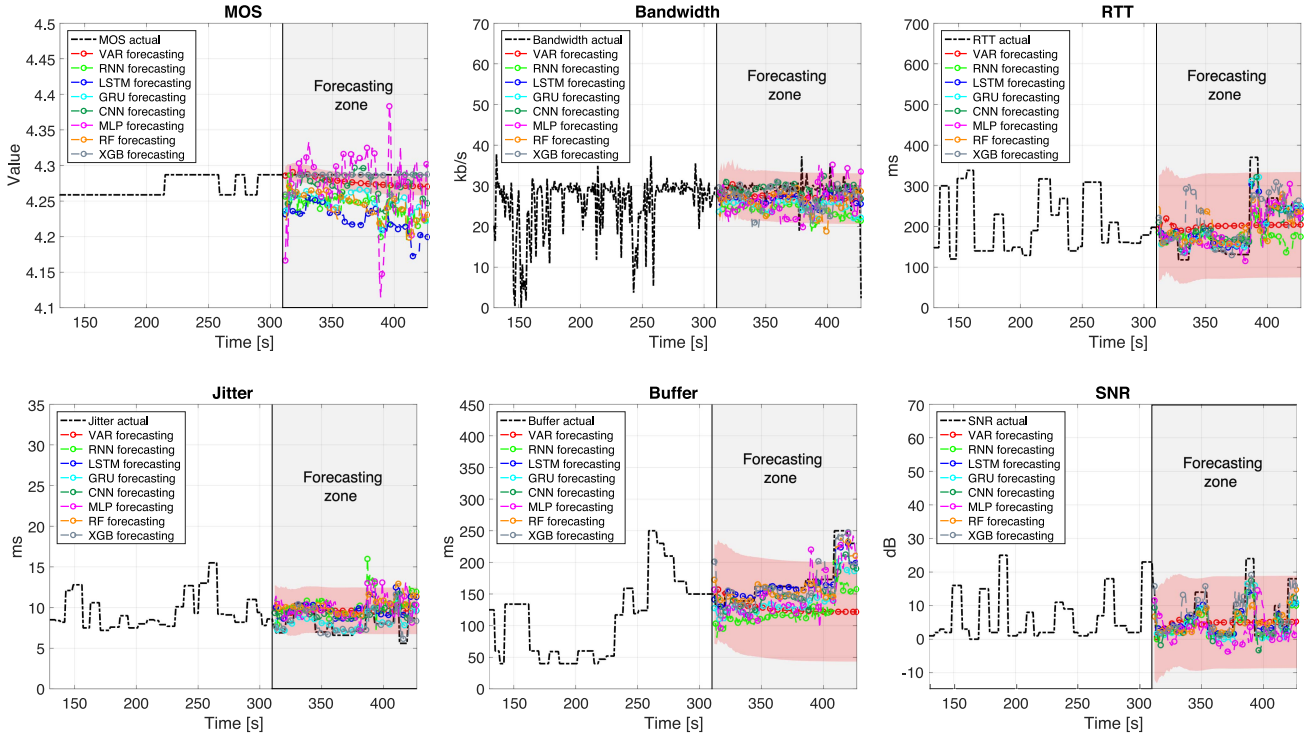


Fig. 10. G.729 codec: Multivariate time series forecasting for: MOS, Bandwidth, RTT, Jitter, Buffer, SNR, along with the 95% forecasting intervals in shaded pale red. In the gray area on the right, the results of forecasting for each technique.

forecasting technique in Table IV, the corresponding MAPE value is indicated in red.

For each voice flow we have repeated a lag analysis (just as seen in Section IV-A), and we have reported the optimal lag value p^* close to VAR model in each sub-table of Table IV.

Let us start to notice some general facts valid for all the experiments. For each voice flow, we can notice that the three performance indicators (RMSE, MAE, and MAPE) exhibit very different ranges for each variable. For instance, in the case of MOS, RMSE and MAE never reach the value 1. This is due to the fact that all the chosen codecs guarantee a good

TABLE IV
PERFORMANCE METRICS (RMSE, MAE, MAPE) PER CODEC (VOICE FLOWS)

Voice Track #1 (G.722 codec)						
VAR(p*=12)	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.017	14.01	10.41	32.52	48.74	11.41
MAE	0.0149	6.32	53.72	14.64	37.7	10.3
MAPE (%)	0.3	*8.0*	16.0	14.0	31.0	68.0
RNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.044	24.78	162.73	26.46	35.8	10.42
MAE	0.038	19.31	58.62	17.7	23.54	9.44
MAPE (%)	0.86	22.8	21.0	21.6	24.4	65.0
LSTM	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.016	14.05	142.21	16.29	33.73	5.73
MAE	0.007	6.78	53.25	7.79	20.13	4.92
MAPE (%)	*0.16*	9.3	22.9	8.8	15.7	39.0
GRU	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.016	14.04	148.67	16.94	24.05	4.04
MAE	0.011	6.82	52.37	7.34	13.25	3.14
MAPE (%)	0.26	9.4	20.5	*7.3*	11.0	*31.0*
CNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.03	14.36	145.84	22.86	27.06	5.88
MAE	0.017	7.26	48.78	11.36	16.97	5.16
MAPE (%)	0.4	9.7	19.0	11.8	14.3	42.0
MLP	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.03	50.32	181.8	33.06	68.10	8.63
MAE	0.02	28.81	81.09	18.14	41.97	6.57
MAPE (%)	0.4	35.0	40.0	19.0	34.0	54.0
RF	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.025	15.15	167.39	29.64	33.39	8.03
MAE	0.023	7.6	53.08	18.5	23.65	6.27
MAPE (%)	0.52	10.0	17.0	22.0	20.0	50.0
XGB	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.017	14.41	160.57	29.28	20.78	6.28
MAE	0.012	6.55	47.1	13.77	11.21	4.46
MAPE (%)	0.27	9.5	*14.5*	13.0	*10.0*	37.0

Voice Track #2 (G.729 codec)						
VAR(p*=13)	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.011	3.57	66.2	21.99	58.1	7.27
MAE	0.01	2.52	56.5	19.41	45.3	5.55
MAPE (%)	0.2	17.0	32.0	27.0	24.0	165.0
RNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.04	6.21	64.4	28.4	52.5	5.03
MAE	0.04	5.69	48.9	25.5	49.8	2.37
MAPE (%)	1.0	26.0	23.0	35.0	29.0	*35.0*
LSTM	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.065	3.97	33.69	21.8	14.8	4.58
MAE	0.063	3.04	22.19	19.2	9.43	3.21
MAPE (%)	1.4	18.0	11.0	26.0	*4.9*	90.0
GRU	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.04	5.24	31.36	11.5	35.8	4.71
MAE	0.038	4.62	18.7	8.25	33.17	3.22
MAPE (%)	0.9	22.0	*10.0*	11.0	19.0	69.0
CNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.024	3.25	37.65	18.96	27.6	4.63
MAE	0.018	1.8	30.31	16.3	22.36	3.21
MAPE (%)	0.4	*14.0*	17.0	23.0	12.0	80.0
MLP	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.05	3.59	57.5	22.79	30.62	4.61
MAE	0.05	2.19	49.8	20.23	25.7	3.44
MAPE (%)	1.1	17.0	30.0	28.0	15.0	118.0
RF	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.05	5.14	45.65	27.56	17.25	5.41
MAE	0.043	3.87	34.97	25.16	12.41	3.89
MAPE (%)	1.0	21.0	19.0	34.0	7.0	112.0
XGB	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.010	4.57	56.9	9.45	19.75	4.18
MAE	0.004	3.2	37.5	5.75	14.18	2.57
MAPE (%)	*0.1*	17.0	21.0	7.0	8.0	82.0

Voice Track #3 (MPEG-16 codec)						
VAR(p*=11)	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.006	3.97	349.9	34.07	63.12	9.17
MAE	0.003	2.84	105.5	19.8	43.4	7.86
MAPE (%)	0.07	9.7	23.0	11.0	44.0	94.0
RNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.007	5.18	325.5	75.8	68.7	12.4
MAE	0.006	4.08	154.4	63.3	62.4	11.1
MAPE (%)	0.14	14.0	64.0	38.0	113.0	65.0
LSTM	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.006	4.23	229.15	32.79	41.76	6.31
MAE	0.003	2.75	67.1	20.83	18.83	3.55
MAPE (%)	0.07	*9.0*	16.4	12.0	14.0	88.0
GRU	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.004	4.39	235.75	27.1	34.52	5.79
MAE	0.003	3.1	74.5	15.08	18.46	3.97
MAPE (%)	0.06	9.6	20.0	8.0	15.0	79.0
CNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.0045	4.35	198.99	25.54	21.4	5.82
MAE	0.0026	2.86	66.4	14.45	13.45	3.79
MAPE (%)	0.061	9.4	23.0	6.7	15.0	72.0
MLP	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.004	4.86	231.6	42.13	39.4	8.25
MAE	0.0025	3.33	74.3	21.3	23.4	2.7
MAPE (%)	0.06	11.0	21.0	12.0	22.0	*49.0*
RF	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.006	4.38	332.7	30.5	23.3	5.93
MAE	0.002	2.8	93.1	18.6	15.2	4.15
MAPE (%)	0.05	9.0	17.0	10.0	17.0	54.0
XGB	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.007	4.51	270.72	22.6	17.75	7.32
MAE	0.0019	3.01	60.1	1.24	8.91	4.62
MAPE (%)	*0.04*	9.8	*11.0*	5.0	9.2	58.0

Voice Track #4 (OPUS codec)						
VAR(p*=11)	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.02	3.2	22.2	67.9	78.1	6.67
MAE	0.02	1.58	18.2	60.1	77.3	5.01
MAPE (%)	0.4	3.1	15.0	13.0	177.0	48.0
RNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.04	3.25	37.8	201.1	40.35	5.9
MAE	0.04	1.62	35.8	198.1	38.9	4.9
MAPE (%)	0.8	3.2	28.0	42.0	90.0	61.0
LSTM	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.016	3.25	8.05	27.97	16.73	3.96
MAE	0.016	1.53	6.4	22.8	15.63	2.78
MAPE (%)	0.37	3.2	5.0	5.0	36.0	*32.0*
GRU	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.005	3.19	19.36	17.7	36.5	5.57
MAE	0.005	1.3	18.6	12.6	35.6	4.8
MAPE (%)	0.1	*3.0*	15.0	*2.8*	82.0	58.0
CNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.008	3.44	10.1	23.9	13.1	4.38
MAE	0.008	1.84	8.25	19.6	11.5	3.32
MAPE (%)	0.2	4.0	6.0	4.0	26.0	42.0
MLP	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.01	4.6	10.4	34.3	23.05	5.03
MAE	0.01	3.1	8.32	29.3	21.37	4.09
MAPE (%)	0.2	6.0	6.0	6.0	48.0	50.0
RF	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.014	3.05	8.32	69.3	19.07	4.5
MAE	0.013	1.5	6.1	54.2	16.4	2.97
MAPE (%)	0.3	3.0	4.8	11.0	38.0	35.0
XGB	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.004	3.4	6.5	47.4	6.7	5.1
MAE	0.003	1.64	4.4	35.5	5.5	3.34
MAPE (%)	*0.07*	3.4	*3.5*	7.0	*12.0*	33.0

Voice Track #5 (GSM codec)						
VAR(p*=5)	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.06	3.7	37.5	7.6	60.0	10.6
MAE	0.06	1.5	21.5	6.3	42.8	3.3
MAPE (%)	1.3	25.0	13.0	8.0	38.0	170.0
RNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.06	12.8	46.8	22.0	35.7	9.26
MAE	0.05	11.8	43.2	21.6	26.5	8.3
MAPE (%)	1.0	40.0	30.0	29.0	30.0	260.0
LSTM	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.06	3.8	35.1	4.4	24.4	5.4
MAE	0.06	1.9	20.5	3.4	14.4	3.9
MAPE (%)	1.3	25.0	12.0	4.0	15.0	80.0
GRU	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.04	3.8	33.5	4.1	20.4	4.9
MAE	0.03	2.03	20.7	3.1	1.4	2.8
MAPE (%)	0.8	25.0	13.0	4.0	*10.0*	*48.0*
CNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.021	4.6	22.4	3.48	24.3	4.8
MAE	0.018	3.0	15.1	2.72	11.5	3.05
MAPE (%)	0.4	30.0	8.0	3.7	11.0	70.0
MLP	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.06	4.1	32.3	3.4	21.6	4.9
MAE	0.04	2.3	19.2	2.5	10.6	3.1
MAPE (%)	0.9	25.0	12.0	3.65	11.0	70.0
RF	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.06	3.7	36.3	6.0	42.2	5.1
MAE	0.05	1.4	21.4	4.6	27.9	3.3
MAPE (%)	1.0	*20.0*	13.0	6.0	24.0	65.0
XGB	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.04	3.7	30.9	4.5	28.2	4.79
MAE	0.01	1.49	14.2	2.4	17.1	2.9
MAPE (%)	*0.2*	25.0	*7.0*	*3.6*	23.0	50.0

Voice Track #6 (SPX8000 codec)						
VAR(p*=8)	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.02	4.9	108.4	30.2	123.3	8.7
MAE	0.014	3.9	56.8	18.5	109.2	6.4
MAPE (%)	0.3	13.0	23.0	17.0	58.0	26.0
RNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.06	5.14	110.0	47.6	110.7	8.01
MAE	0.06	4.2	50.2	40.1	96.4	6.34
MAPE (%)	1.4	14.0	17.0	40.0	50.0	28.0
LSTM	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.016	4.9	80.8	16.7	63.7	6.5
MAE	0.013	3.8	30.8	10.5	45.8	4.6
MAPE (%)	0.3	*12.0*	*14.0*	10.0	20.0	20.0
GRU	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.01	5.01	68.6	17.4	49.7	7.1
MAE	0.008	4.04	41.3	10.4	40.5	5.2
MAPE (%)	0.2	13.0	20.0	9.6	20.0	26.0
CNN	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.02	5.26	55.01	13.3	26.3	6.78
MAE	0.02	4.14	33.2	8.6	18.4	5.3
MAPE (%)	0.2	13.0	17.0	*9.0*	*9.7*	26.0
MLP	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.02	5.1	51.6	15.3	35.4	7.2
MAE	0.014	4.14	31.26	9.9	27.1	5.7
MAPE (%)	0.3	13.0	16.0	11.0	15.0	26.0
RF	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.02	5.2	95.5	21.9	80.7	6.8
MAE	0.02	4.01	54.8	15.1	66.1	5.6
MAPE (%)	0.4	13.0	24.0	16.0	33.0	26.0
XGB	**MOS**	**BW**	**RTT**	**Jitter**	**Buffer**	**SNR**
RMSE	0.01	5.6	100.9	23.6	79.8	6.8
MAE	0.006	4.4	54.5	13.3	66.3	4.4
MAPE (%)	*0.1*	14.0	20.0	13.0	30.0	*18.0*

perceived quality, with a MOS varying within a limited range of values (MOS values never lie below 4). This directly reflects into low values of all the indicators.

Conversely, RTT varies within a great range of values with some unusual peaks due to the temporary network conditions (see, e.g., the peaks of RTT in green at about $t = 30$ s and $t = 415$ s in Fig. 2). In such cases, all the forecasting techniques are (obviously) not able to predict this behavior, thus the prediction error is quite large. Such a condition directly reflects onto performance indicators and, in particular, onto RMSE whose values are high and hugely different between them since RMSE tends to magnify large errors. Yet, the SNR exhibits a quite standard smooth behavior with weak oscillations and not unusual peaks. Thus, the performance indicators are not dramatically high, indicating a satisfying forecasting accuracy.

Indeed, if we compare the performance accuracy for each technique, we can observe that: VAR technique, as also mentioned before, could return satisfactory result when the time series does not exhibit too many non-linearities; on average, deep techniques such as LSTM, GRU, and CNN produce satisfactory results (see MAPE values in red). For LSTM and GRU the results are justified thanks to the presence of a memory-based internal structure able to keep track of past values. For CNN, results are justified by the presence of a convolutional

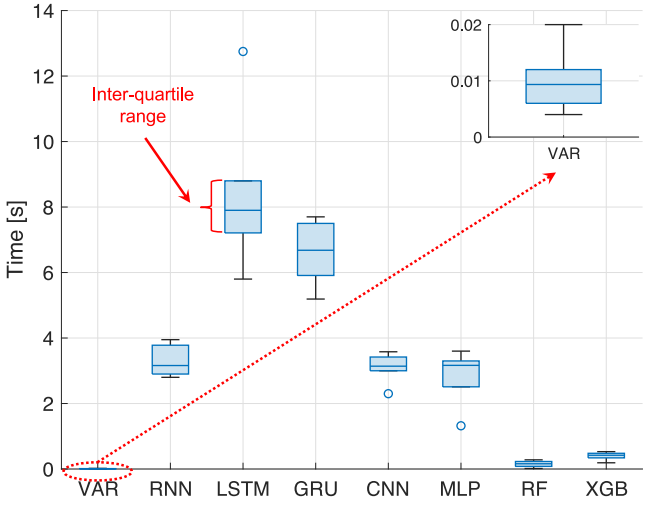


Fig. 11. Box-plots of the elapsed times for each forecasting technique applied to all the available voice flows.

structure able to derive the most significant temporal features. Remaining deep based techniques (RNN and MLP) exhibit less performing accuracy results due to their naive internal structure which does not exploit any particular characteristic of temporal data. Among standard machine learning techniques, XGBoost exhibits the best performance since it relies on a combination of ensemble models useful to improve the quality of prediction.

The performance analysis should be complemented by a time evaluation to better compare the considered techniques. Figure 11 shows the results of such a comparison obtained by exploiting a platform equipped with 2 virtual CPUs (Intel Xeon @2.30 GHz) and 13 GB of RAM. In this time analysis we take into account all the experiments (namely, all the voice flows) in order to highlight possible variability through a boxplot representation. First, such an analysis reveals that the VAR technique is able to perform the forecasting in few milliseconds as shown within the top-right inset. This is basically due to the fact that VAR is built through a linear combination of lagged values to forecast the next sample. In contrast, deep methods require more time (in particular during the training time) to perform the forecasting due to their internal structure which can be more or less complicated (e.g., memory-based cells, convolution operations). In the middle we find XGB and RF which, relying on an optimized tree-based structure, are quite fast. The boxplot representation also highlights that, when using a technique with a complex internal structure (typically, deep-based techniques) the time variability directly increases. This is obviously connected to the fact that a more complex structure may produce higher delays. This behavior is captured through the inter-quartile range (IQR) defined as the difference between third and first quartiles. Large IQR value imply more dispersed values. In case of deep-based techniques we observe the following IQR values: RNN (0.88), LSTM (1.59), GRU (1.58), CNN (0.42), MLP (0.79). In case of standard learning techniques we have: RF (0.15), XGB (0.14). Finally, VAR is the more stable having the smallest IQR value amounting to 0.006.

A. Main Findings

Through the proposed assessment we are able to infer some general considerations about the evaluated forecasting techniques. First of all, we can reasonably say that there is not a definitive winner, since the forecasting complexity does not allow to select an outperforming technique in an absolute sense. An insightful comparison can be made between the statistical approach (represented by VAR) and the learning techniques.

First, we highlight that the VAR method allows a complete control on the analytical structure of each time series. In particular, its ancillary analyses (e.g., residuals, impulse response) provide deep insights about the time series composition and their mutual relationships. In contrast, the data-driven approach adopted by learning techniques does not allow to capture many analytical details. For instance, guaranteeing the stationarity condition (not required by learning approaches) allows to obtain useful descriptors (mean, variance, correlation) of the future behavior of a time series. Conversely, in case the stationarity condition is violated (namely if the series is consistently increasing over time) the sample mean and variance will grow with the sample size, and will tend to underestimate the mean and variance in future periods.

Second, VAR is a good choice in case the time series exhibits a good stability over time, or when the observation time is wider than tens of minutes (e.g., per-month or per-year). In this latter case, in fact, the temporal irregularities tend to be smoother, and the linear combination of past lagged values offer better performance. Conversely, being intrinsically adaptive, learning techniques are more responsive in presence of network parameters fluctuations. Furthermore, VAR offers challenging performance in terms of compute times due to the simplicity of the model. On the other hand, deep recurrent methods (RNN, LSTM, GRU, CNN, MLP) exhibit slower computation times along with high temporal uncertainty (high IQR values) mainly due to the complex internal structure. Among standard ML techniques, XGBoost offers an interesting trade-off between accuracy and time.

We finally notice that, differently from all the learning techniques, VAR does not need any hyper-parameter tuning (other than the optimal lag) which, if not accurate, could lead to poor performance.

VI. Conclusion

In this work we tackle the problem of forecasting mobile VoIP traffic in a real cellular environment. The main purpose is to provide precious information to network operators allowing them to optimize the network planning in mobile environments. In particular, we characterize the temporal evolution of the most important QoS/QoE descriptors of VoIP traffic through a multivariate time series assessment. Forecasting techniques such as Vector Autoregression and machine learning approaches have been compared to highlight *pros* and *cons* both in terms of performance and times.

The work presents a series of novelty elements. First, we propose a multivariate time series characterization of network descriptors, an approach currently used by econometricians to

model and predict the market stock evolution. Through such an approach it is possible to analytically capture the interdependencies among the stochastic processes which govern the network variables behavior. Then, the time series problem has been turned into a supervised learning framework through the sliding window technique.

Such reframing of the problem is useful to: *i*) reinterpret the classic concepts of training/test sets in terms of temporal values of a time series aimed at forecasting future values of network descriptors; *ii*) compare in a critical manner statistical techniques (here represented by the VAR model) and machine learning methods. Results show that VAR is the optimal choice when a complete analytical control on the variables is needed, when the network fluctuations are not so persistent, or when strict elaboration time constraints are present. In contrast, learning-based techniques provide excellent accuracy in case of network instability due their data-driven approach.

Finally, the whole assessment is supported by an experimental campaign in a real mobility LTE-A environment, where through the evolved RTCP-XR protocol, we are able to derive network metrics typically neglected in the literature (e.g., MOS, SNR, playout delay buffer).

Such a work remains open for future investigations along several directions: *i*) the main techniques adopted for this analysis could be extended to technologies such as 5G as they become more pervasive and with the possibility of acquiring data from real settings; *ii*) many derived models could be used as benchmark to design more realistic network simulators; *iii*) new parameters such as the car's speed could be gathered and related to the behavior of the VoIP metrics; *iv*) it could be possible to repeat the whole analysis in a transformed domain (e.g., wavelet domain in place of time domain).

References

[1] M. L. Marí-Altozano, S. S. Mwanje, S. Luna-Ramírez, M. Toril, H. Sanneck, and C. Gijón, "A service-centric Q-learning algorithm for mobility robustness optimization in LTE," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 3, pp. 3541–3555, Sep. 2021.

[2] D. Minovski, N. Ögren, C. Åhlund, and K. Mitra, "Throughput prediction using machine learning in LTE and 5G networks," *IEEE Trans. Mobile Comput.*, vol. 22, no. 3, pp. 1825–1840, Mar. 2023.

[3] "Ericsson mobility report—Jun. 2021." Ericsson. 2021. [Online]. Available: https://www.ericsson.com/4a03c2/assets/local/mobility-report/documents/2021/june-2021-ericsson-mobility-report.pdf

[4] *5G Non-Standalone to 5G Standalone*, Nokia Corp., Espoo, Finland, 2021.

[5] E. Zeydan, J. Mangues-Bafalluy, O. Dedeoglu, and Y. Turk, "Performance comparison of QoS deployment strategies for cellular network services," *IEEE Access*, vol. 8, pp. 176073–176088, 2020.

[6] P. Uthansakul, P. Anchuen, M. Uthansakul, and A. A. Khan, "QoE-aware self-tuning of service priority factor for resource allocation optimization in LTE networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 887–900, Jan. 2020.

[7] M. Di Mauro and A. Liotta, "An experimental evaluation and characterization of VoIP over an LTE-A network," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 3, pp. 1626–1639, Sep. 2020.

[8] J. Wszolek, S. Ludyga, W. Anzel, and S. Szott, "Revisiting LTE LAA: Channel access, QoS, and coexistence with WiFi," *IEEE Commun. Mag.*, vol. 59, no. 2, pp. 91–97, Feb. 2021.

[9] C. Bhar, N. Chatur, A. Mukhopadhyay, G. Das, and D. Datta, "Designing a green optical network unit using ARMA-based traffic prediction for quality of service-aware traffic," *Photon. Netw. Commun.*, vol. 32, pp. 407–421, Nov. 2016.

[10] A. Sang and S.-Q. Li, "A predictability analysis of network traffic," in *Proc. INFOCOM*, 2000, pp. 342–351.

[11] Z. Tian and F. Li, "Network traffic prediction method based on autoregressive integrated moving average and adaptive volterra filter," *Int. J. Commun. Syst.*, vol. 34, no. 12, p. e4891, 2021, doi: 10.1002/dac.4891.

[12] H. Yang, X. Li, W. Qiang, Y. Zhao, W. Zhang, and C. Tang, "A network traffic forecasting method based on SA optimized ARIMA–BP neural network," *Comput. Netw.*, vol. 193, pp. 1–12, Jul. 2021.

[13] N. Bui and J. Widmer, "Data-driven evaluation of anticipatory networking in LTE networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 10, pp. 2252–2265, Oct. 2018.

[14] V. B. Dharmadhikari and J. D. Gavade, "An NN approach for MPEG video traffic prediction," in *Proc. ICSTE*, 2012, pp. 57–61.

[15] A. Y. Nikravesh, S. A. Ajila, C.-H. Lung, and W. Ding, "Mobile network traffic prediction using MLP, MLPWD, and SVM," in *Proc. IEEE BigData Congr.*, 2016, pp. 402–409.

[16] C. Gijón, M. Toril, S. Luna-Ramírez, M. L. Marí-Altozano, and J. M. Ruiz-Aviles, "Long-term data traffic forecasting for network dimensioning in LTE with short time series," *Electronics*, vol. 10, no. 10, pp. 1–19, 2021.

[17] D. Andreoletti, S. Troia, F. Musumeci, S. Giordano, G. Maier, and M. Tornatore, "Network traffic prediction based on diffusion convolutional recurrent neural networks," in *Proc. INFOCOM*, 2019, pp. 246–251.

[18] S. P. Sone, J. J. Lehtomaki, and Z. Khan, "Wireless traffic usage forecasting using real enterprise network data: Analysis and methods," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 777–797, 2020.

[19] E. Ak and B. Canberk, "Forecasting quality of service for next-generation data-driven WiFi6 campus networks," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 4, pp. 4744–4755, Dec. 2021.

[20] A. Azzouni and G. Pujolle, "NeuTM: A neural network-based framework for traffic matrix prediction in SDN," in *Proc. NOMS*, 2018, pp. 1–5.

[21] J. Zhou, D. Ding, Z. Wu, and Y. Xiu, "Spatial context-aware time-series forecasting for QoS prediction," *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 2, pp. 918–931, Jun. 2023, doi: 10.1109/TNSM.2023.3250512.

[22] A. P. Wibawa, A. B. P. Utama, H. Elmunsyah, U. Pujianto, F. A. Dwiyanto, and L. Hernandez, "Time-series analysis with smoothed convolutional neural network," *J. Big Data*, vol. 9, no. 44, pp. 1–18, 2022.

[23] J. F. Torres, D. Hadjout, A. Sebaa, F. Martínez-Álvarez, and A. Troncoso, "Deep learning for time series forecasting: A survey," *Big Data*, vol. 9, no. 1, pp. 3–21, 2021.

[24] K. Abidi, E. Hajlaoui, and M. Abdellaoui, "Handover simulation of LTE and LTE-A standards," in *Proc. SoCPaR*, 2014, pp. 157–162.

[25] G. Carullo, M. Tambasco, M. Di Mauro, and M. Longo, "A performance evaluation of WebRTC over LTE," in *Proc. WONS*, 2016, pp. 1–6.

[26] H.-F. Bermudez, R. Sanchez-Iborra, J. L. Arciniegas, W. Y. Campo, and M.-D. Cano, "Performance validation of NS3-LTE emulation for live video streaming under QoS parameters," in *Proc. WiMob*, 2017, pp. 300–307.

[27] M. Sahu and A. A. Kherani, "End-to-end delay jitter in LTE uplink: Simple models, empirical validation & applications," in *Proc. COMSNETS*, 2020, pp. 221–228.

[28] A. Stornig, A. Fakhreddine, H. Hellwagner, P. Popovski, and C. Bettstetter, "Video quality and latency for UAV teleoperation over LTE: A study with NS3," in *Proc. VTC*, 2021, pp. 1–7.

[29] M. L. Marí-Altozano, S. Luna-Ramírez, M. Toril, and C. Gijón, "A QoE-driven traffic steering algorithm for LTE networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11271–11282, Nov. 2019.

[30] E. Cipressi and M. L. Merani, "An effective machine learning (ML) approach to quality assessment of voice over IP (VoIP) calls," *IEEE Netw. Lett.*, vol. 2, no. 2, pp. 90–94, Jun. 2020.

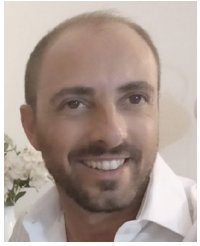
[31] J. O. Olaifa and D. Arifler, "Using system-level simulation to evaluate downlink throughput performance in LTE-A networks with clustered user deployments," in *Proc. IWSLS*, 2016, pp. 1–6.

[32] S. Park, M. Agiwal, H. Kwon, and H. Jin, "An evaluation methodology for spectrum usage in LTE-A networks: Traffic volume and resource utilization perspective," *IEEE Access*, vol. 7, pp. 67863–67873, 2019.

[33] I. Oussakel, P. Owezarski, and P. Berthou, "Experimental estimation of LTE-A performance," in *Proc. CNSM*, 2019, pp. 1–9.

[34] J. Qu, F. Liu, Y. Ma, and J. Fan, "Temporal-spatial collaborative prediction for LTE-R communication quality based on deep learning," *IEEE Access*, vol. 8, pp. 94817–94832, 2020.

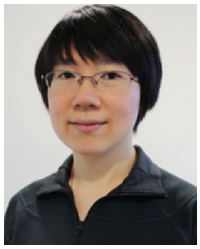
[35] U. S. Hashmi et al., "Towards real-time user QoE assessment via machine learning on LTE network data," in *Proc. VTC*, 2019, pp. 1–7.
[36] H. Elsherbiny, H. M. Abbas, H. Abou-Zeid, H. S. Hassanein, and A. Noureldin, "4G LTE network throughput modelling and prediction," in *Proc. GLOBECOM*, 2020, pp. 1–6.
[37] V. Raida, P. Svoboda, and M. Rupp, "Real world performance of LTE downlink in a static dense urban scenario—An open dataset," in *Proc. GLOBECOM*, 2020, pp. 1–6.
[38] "Cellmapper." Accessed: Jul. 27, 2023. [Online]. Available: http://www.cellmapper.net
[39] "Linphone." Accessed: Jul. 27, 2023. [Online]. Available: http://www.linphone.org
[40] "RTP control protocol extended reports (RTCP XR)," IETF, RFC 3611, 2003. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc3611
[41] "Wireshark." Accessed: Jul. 27, 2023. [Online]. Available: http://wireshark.org
[42] *The E-Model: A Computational Model for Use in Transmission Planning, in Force*, Rec. G.107, ITU-T, Geneva, Switzerland, 2015. [Online]. Available: https://www.itu.int/rec/T-REC-G.107
[43] "Cisco prime collaboration assurance - advanced and analytics guide, 12.1 service pack 3 engineering special 4 - Perform diagnostics." Accessed: Jul. 27, 2023. [Online]. Available: https://www.cisco.com/c/en/us/support/cloud-systems-management/prime-collaboration/products-user-guide-list.html
[44] 2023, "Multivariate time series characterization and forecasting of VoIP traffic in real mobile networks," IEEE DataPort, doi: 10.21227/jef5-4w68.
[45] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Proc. ISIT*, 1973, pp. 267–281.
[46] H. Mumtaz and O. Rummel, "Recent developments in structural VAR modelling," in *Centre for Central Banking Studies*. London, U.K.: Bank of England, 2015.
[47] R. J. Hyndman and G. Athanasopulos, *Forecasting: Principles and Practice*. Melbourne, VIC, Australia: Otexts Publ., 2018.
[48] L. G. Godfrey, "Testing for higher order serial correlation in regression equations when the regressors include lagged dependent variables," *Econometrica*, vol. 46, pp. 1303–1313, Nov. 1978.
[49] L. Kilian and H. Lütkepohl, *Structural Vector Autoregressive Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2017.
[50] D. Edgerton and G. Shukur, "Testing autocorrelation in a system perspective testing autocorrelation," *Econometric Rev.*, vol. 18, no. 4, pp. 343–386, 1999.
[51] A. Hatemi-J, "Multivariate tests for autocorrelation in the stable and unstable VAR models," *Economic Model.*, vol. 21, no. 4, pp. 661–683, 2004.
[52] R. Tsay, *Multivariate Time Series Analysis: With R and Financial Applications*. Hoboken, NJ, USA: Wiley, 2014.
[53] D. A. Dickey and W. A. Fuller, "Distribution of the estimators for autoregressive time series with a unit root," *J. Amer. Stat. Assoc.*, vol. 74, no. 366, pp. 427–431, 1979.
[54] J. D. Hamilton, *Time Series Analysis*. Princeton, NJ, USA: Princeton Univ. Press, 1994.
[55] H. Lütkepohl, *New Introduction to Multiple Time Series Analysis*. Berlin, Germany: Springer-Verlag, 2005.
[56] W. Ploberger and W. Krämer, "The Cusum test with Ols residuals," *Econometrica*, vol. 60, no. 2, pp. 271–285, 1992.
[57] E. Zivot and J. Wang, "Vector models for multivariate time series," in *Modeling Financial Time Series with S-Plus*. New York, NY, USA: Springer, 2003, pp. 369–413.
[58] C. Sims, "Macroeconomics and reality," *Econometrica*, vol. 48, pp. 1–48, Jan. 1980.
[59] H. Lütkepohl and D. S. Poskitt, "Estimating orthogonal impulse responses via vector autoregressive models," *Econometric Theory*, vol. 7, no. 4, pp. 487–496, 1991.
[60] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.
[61] R. P. Masini, M. C. Medeiros, and E. F. Mendes, "Machine learning advances for time series forecasting," *J. Econ. Surv.*, vol. 37, no. 1, pp. 76–111, 2023.
[62] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[63] K. Cho et al., "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. EMNLP*, 2014, pp. 1724–1734.
[64] A. Essien and C. Giannetti, "A deep learning model for smart manufacturing using convolutional LSTM neural network autoencoders," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 6069–6078, Sep. 2020.
[65] P. Hill, J. Biggs, V. Ponce-López, and D. Bull, "Time-series prediction approaches to forecasting deformation in sentinel-1 InSAR data," *J. Geoph. Res. Solid Earth*, vol. 126, no. 3, pp. 1–17, 2021.
[66] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.



Mario Di Mauro (Senior Member, IEEE) received the Ph.D. degree in information engineering from the University of Salerno, Italy, in 2018. From 2007 to 2012, he was with Research Consortium on Telecommunications (formerly Ericsson Lab Italy) as an Industrial Researcher. He is an Assistant Professor of Telecommunications with the University of Salerno. His main fields of interest include: network availability and security and data analysis for telecommunication infrastructures.

Giovanni Galatro received the Laurea degree (summa cum laude) in information engineering from the University of Salerno, Italy, in 2018. In 2017, he got a scholarship with Telecommunication and Applied Statistics Groups, focused on the analysis of modern telecommunication infrastructures. He is actually a Cloud Engineer with IBM.

Fabio Postiglione received the Ph.D. degree in information engineering from the University of Salerno, Italy, in 2005, where he is currently an Associate Professor of Statistics. He has coauthored over 120 papers, mainly published in international journals. His main research interests include degradation analysis, lifetime estimation, reliability and availability evaluation of complex systems (telecommunication networks, fuel cells), Bayesian statistics, and data analysis.



Wei Song (Member, IEEE) received the Ph.D. degree in computer science from the Queensland University of Technology, Brisbane, QLD, Australia, in 2012, where she was a Research Fellow from 2012 to 2015. In 2016, she became a Professor with Shanghai Ocean University, China, where she is the Co-Founder of Joint Intellisensing Lab. Her research interests include image/video processing for underwater computer vision, big data analysis, remote sensing image analysis, and its application in marine fields. She was rewarded as "Eastern Scholar" by the Shanghai Institutions of Higher Learning, Shanghai, in 2016.

Antonio Liotta (Senior Member, IEEE) is a Full Professor of Data Science with the Free University of Bozen-Bolzano, Italy. His team is at the forefront of influential research in data science and artificial intelligence, specifically in the context of smart cities, IoT, and smart sensing. He is renowned for his contributions to miniaturized machine learning and artificial neural networks acceleration via network science methods. He has over 350 publications to his credit. He is the Editor-in-Chief of the Internet of Things book series (Springer).

Open Access funding provided by 'Università degli Studi di Salerno' within the CRUI CARE Agreement