Neural Networks 71 (2015) 204–213

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Prediction of telephone calls load using Echo State Network with exogenous variables

Filippo Maria Bianchi [a,*], Simone Scardapane [a], Aurelio Uncini [a], Antonello Rizzi [a], Alireza Sadeghian [b]

[a] Department of Information Engineering, Electronics and Telecommunications (DIET), "Sapienza" University of Rome, Via Eudossiana 18, 00184 Rome, Italy

[b] Department of Computer Science, Ryerson University, 350 Victoria Street, Toronto, ON M5B 2K3, Canada

ARTICLE INFO

Article history:
Received 5 June 2015
Received in revised form 23 July 2015
Accepted 28 August 2015
Available online 7 September 2015

Keywords:
Time-series
Forecasting
Echo State Networks
Exogenous variables
Genetic algorithm
Call data records

ABSTRACT

We approach the problem of forecasting the load of incoming calls in a cell of a mobile network using Echo State Networks. With respect to previous approaches to the problem, we consider the inclusion of additional telephone records regarding the activity registered in the cell as exogenous variables, by investigating their usefulness in the forecasting task. Additionally, we analyze different methodologies for training the readout of the network, including two novel variants, namely ν-SVR and an elastic net penalty. Finally, we employ a genetic algorithm for both the tasks of tuning the parameters of the system and for selecting the optimal subset of most informative additional time-series to be considered as external inputs in the forecasting problem. We compare the performances with standard prediction models and we evaluate the results according to the specific properties of the considered time-series.

1. Introduction

Time-Series Forecasting (TSF) refers to the problem of predicting future values of a time-series (TS), starting from a previously observed history (De Gooijer & Hyndman, 2006). In this paper, we are concerned specifically with the TSF problem of telephone activity loads. This is closely related to the forecasting of workload in call centers (Aksin, Armony, & Mehrotra, 2007) where, usually, only the TS containing the load of incoming calls is taken into account and the other external variables considered for the prediction usually possess a very different nature (e.g. advertisement, catalogs, calendar effects Andrews & Cunningham, 1995; Antipov & Meade, 2002; Soyer & Tarimcilar, 2008). An accurate Short-Term Load Forecast (STLF) method would save operating costs, keep power markets efficient and provide a better understanding of the dynamics of the observed system. On the other hand, a wrong prediction could cause either a load overestimation, which leads to the excess of reserving resources and consequently more costs and contract curtailments for market participants, or a load underestimation resulting in failures in providing enough reserves, thereby more costly ancillary services (Bunn, 2000; Ruiz & Gross, 2008).

Specifically, in this work we treat the problem of STFL relative to the telephonic activities registered on a cell covered by an antenna of a mobile phone network. Relatively to each cell there are different kinds of data that describes the volume and the number of both outgoing and incoming calls, from which we generate different TSs. Our work is focused on forecasting the values of a specific TS using past measurements and leveraging on the information contained in the remaining TSs, considered as exogenous variables which are presented as input to the system along with the TS that must be predicted. In particular, in this work we consider call records collected in the Orange telephone dataset published for the "Data for Development" (D4D) challenge (Blondel et al., 2012). More information on the TSs and how they are generated in a pre-processing phase is provided in Section 3.

As forecast method we use a standard Echo State Network (ESN) (Butcher, Verstraeten, Schrauwen, Day, & Haycock, 2013; Jaeger & Haas, 2004; Lukoševičius & Jaeger, 2009; Verstraeten, Schrauwen, d'Haene, & Stroobandt, 2007), which is a particular class of Recurrent Neural Network (RNN). The main peculiarity of ESNs is that the recurrent part of the network (the *reservoir*) is considered fixed, and only a non-recurrent part (termed *readout*) is

* Corresponding author. Tel.: +39 06 44585495; fax: +39 06 4873300.
E-mail addresses: filippomaria.bianchi@uniroma1.it (F.M. Bianchi), simone.scardapane@uniroma1.it (S. Scardapane), aurelio.uncini@uniroma1.it (A. Uncini), antonello.rizzi@uniroma1.it (A. Rizzi), asadeghi@ryerson.ca (A. Sadeghian).

http://dx.doi.org/10.1016/j.neunet.2015.08.010
0893-6080/

effectively trained. In this way, it is possible to use standard linear regression routines for solving the overall optimization problem, without the need of complex backpropagation of the errors over the dynamic portion of the network (Pearlmutter, 1995).

The ESN has proven to be a very effective methodology for the STLF problem in a wide range of applicative domains, including the forecasting of the water inflow to an hydropower plant reservoir (Sacchi, Ozturk, Principe, Carneiro, & da Silva, 2007), electric load in power systems (Deihimi & Showkati, 2012; Qingsong, Xiangmo, Zuren, Yisheng, & Baohua, 2011; Showkati, Hejazi, & Elyasi, 2010), price prediction in economic markets (Lin, Yang, & Song, 2009) and hourly wind speeds (Ferreira, Ludermir, de Aquino, Lira, & Neto, 2008). On the other hand, because of the property of 'short-term memory', ESN is not suitable for long-term predictions (Peng, Lei, Li, & Peng, 2014); for longer forecasting horizons, very basic averaging models outperforms many more complex alternatives (Taylor, 2008). The traditional methods often encounter difficulty in properly modeling the complex nonlinear relationships between load and various exogenous factors, in particular the stochastic ones that influence the load. Conversely, methods based on an ESN model have the ability to learn or map those nonlinear relationships so that, if they are properly constructed and they suitably leverage on historical data, the accuracy of the predicted results can be very high.

In this work we approach the TSF problem with an ESN, facing a novel application concerning the call loads registered in a cell of a mobile telephone network. We use an automatic procedure for identifying the set of exogenous variables during the optimization phase, used as a support information in the prediction. In fact, unlike external variables commonly used in forecasting problems (such as the ambient temperature in the prediction of the electric load in a distribution network), the correlations between the different TSs are not obvious. For training the ESN we consider several approaches in our application, including least-square regression, an elastic net penalty and two versions of the ν-SVR (Shi & Han, 2007). To the best of our knowledge, the last two methods were never investigated for the task of ESN training. We evaluate their performances in the experimental section, where the prediction accuracies are compared with the performances on the same problem by standard forecasting methods, namely the classic Box–Jenkins Auto-Regressive Integrated Moving Average (ARIMA) (Anderson, 1976), Auto-Regressive Integrated Moving Average with Exogenous inputs (ARIMAX) (Huang, Huang, & Wang, 2005) and Triple Exponential Smoothing (TES) (Kalekar, 2004).
It should be pointed out that the ESN depends on several parameters and a correct tuning is essential in order to obtain good performances from the system. An innovative aspect of this paper is the use of a genetic algorithm for simultaneously identifying the optimal settings of the network, and for selecting the most informative subset of TSs to be considered as exogenous variables in the TSF problem.

The remainder of the paper is organized as follows: in Section 2 we review some related works on the STLF using different methods, among which the ESN. In Section 3 we describe the considered TS, in Section 4 we present the proposed ESN-based forecasting system and in Section 5 we report the results obtained and we compare them with the other systems. Finally, in Section 6 we discuss the conclusions and the future works.

Notation

In this paper, vectors are denoted by boldface lowercase letters, such as $\mathbf{a}$, while matrices are denoted by boldface uppercase letters, such as $\mathbf{A}$. All vectors are assumed column vectors, with $\mathbf{a}^T$ denoting the transpose of $\mathbf{a}$. The ith element of a vector is denoted as a_i, while $\mathbf{a}[n]$ is used for time dependence on the index n. $\|a\|_p$ is used for the L_p-norm of a vector. For $p = 2$ this is the standard Euclidean norm, while for $p = 1$ we have $\|\mathbf{a}\|_1 = \sum_i a_i$. Finally, the spectral radius of a generic matrix $\mathbf{A}$ is $\rho(\mathbf{A}) = \max_i \{|\lambda_i(\mathbf{A})|\}$, where $\lambda_i(\mathbf{A})$ is the ith eigenvector of $\mathbf{A}$.

2. Related works

In this section we report different state-of-the-art works in the literature which are related to the problem and to the methodologies considered in this paper. In particular, we firstly review the works which dealt with the STFL problem considering exogenous variables in Section 2.1. Then, we consider the TSF problem approached with methods based on recurrent neural architectures, with particular emphasis on ESN models, in Section 2.2.

2.1. Load prediction using exogenous variables

Load forecasting is vitally important in the deregulated economy and it has many applications including load switching, contract evaluation, and infrastructure development. A large variety of mathematical methods have been developed for load forecasting and some of them use external variables as a support in the prediction task, for increasing the forecast accuracy. Early forecasting studies relative to STLF of call volumes (Andrews & Cunningham, 1995; Antipov & Meade, 2002) applied Auto-Regressive Moving Average (ARMA) models incorporating exogenous inputs along-side MA and AR variables, using transfer functions to help predict outliers, such as special sales promotion periods, and adding exogenous variables in a multiplicative manner for including advertising response and special calendar effects. The underlying intuition is that the predictable behavior of advertising is embedded in the structure of the applications series, thus it is the unpredictable behavior of advertising that will be responsible for innovations in application. In Ibrahim and L'Ecuyer (2013) the authors propose a stochastic model for predicting the call load, which estimates the load in a given period of the day using the correlations between the load observed in the previous hours, the load observed in the day during the previous weeks and the load observed on a second exogenous TS. They stated that modeling correlation structures in the data is not necessary for long-term forecasts. Thus, it is sufficient for real-life call center managers to base their long-term managerial decisions on historical averages, e.g., fixed-effects models. A similar work on TSF (Aldor-Noiman, Feigin, & Mandelbaum, 2009) consists in using a statistical model for predicting the work load in a call center, which is based on a mixed Poisson process approach that considers the effect of events such as billing on the arrival process. The authors demonstrate how to incorporate them as exogenous variables in the model. Exogenous variables are considered also in Niu, Ji, Xing, and Wang (2012), where the prediction of the daily electricity load is performed including the registered temperature in the model as an external variable. They used a special ESN with a different reservoir for each TS: this allow to better catch the dynamic of each single variable. Authors also propose a method for pruning the output weights which, due to the multiplicity of reservoirs, will be huge and could lead to an over-fitting during the training.

2.2. Forecasting with ESNs

Artificial Neural Networks (ANN) have been intensively applied in time-series analysis for several years and many different methodologies and approaches have been explored in the last decade, relatively to the TSF problem (De Gooijer & Hyndman, 2006; Ghiassi, Saidane, & Zimbra, 2005; Zhang, 2001; Zhang, Patuwo, & Hu, 1998). Recently, the ESN has proven to be one of the most effective typology of ANN to be used in TSF, especially

when the considered TS is chaotic (Li, Han, & Wang, 2012; Shi & Han, 2007). More in general, several works have detailed the superior performances obtained using recurrent NN models with respect to standard prediction methods (Ho, Xie, & Goh, 2002; Ilies et al., 2007). The motivation behind the usage of recurrence is that some patterns may repeat over time. A network which remembers previous inputs or the feedback of previous outputs may have greater success in identifying these time dependent patterns.

In the following, we briefly list the main works related to time-series forecasting with ESNs. In Varshney and Verma (2014) the authors use an ESN for forecasting electricity load and they stated that the commonly used feedforward backpropagation network offers good performance, which could however be improved by using recurrence or by considering past inputs and outputs. Another work that uses ESN in TSF is Deihimi and Showkati (2012), where the authors train a different ESN for predicting each hour of the day. They consider 1-h ahead and 24-h ahead TSF problems. The 24-h forecast is obtained calling recursively the 1-h ahead. The training of each ESN is done considering only a specific hour of the day. As concerns the topic of chaotic TS prediction, Li et al. (2012) proposes an alternative to the Bayesian regression for estimating the regularization parameter of the ESN least-square training criterion. The method uses a Laplacian likelihood function rather than a Gaussian one, which has demonstrated to be more robust to noise and outliers. An hybrid model using ESN in conjunction with ARIMA is proposed in Peng et al. (2014), where they forecast a mobile communication traffic series with multiple seasonality. The TS is decomposed according to a wavelet multi-resolution analysis in a sum of different wavelets, one for each seasonality, and a residual smooth term. Each wavelet is an extremely regular TS with a well-defined seasonality which can be effectively predicted with an ARIMA model. The smooth component, instead, is processed with an ESN, and the final prediction is the result of the integration of all the components. Finally, Scardapane, Dianhui, and Panella (in press) investigate a decentralized training algorithm for ESNs, which is applied to chaotic series prediction over a network of agents. Results are comparable to the centralized case.

3. The analyzed time-series

In this work we analyze a set of TSs generated from the data collected in the Orange telephone dataset published for the Data for Development (D4D) challenge (Blondel et al., 2012), which is an open collection of call data records, containing anonymous calling events of Oranges mobile phone users in Ivory Coast. More information on the challenge is available on the website http://www.d4d.orange.com. The data consist of anonymized mobile phone calls and SMS that have been gathered between December 1, 2011 and April 28, 2012 and they are arranged in four different datasets. The datasets are: (1) antenna-to-antenna traffic on an hourly basis, (2) individual trajectories for 50,000 customers for two week time windows with antenna location information, (3) individual trajectories for 50,000 customers over the entire observation period with sub-prefecture location information, and (4) a sample of communication graphs for 5000 customers.

The data we are interested in are the ones coming from the first dataset, i.e. the records relative to antenna-to-antenna traffic. Each record has the following structure:

$$\langle DateTime, ID_a, ID_b, NumCalls, TotTime \rangle$$

where *DateTime* is the time (with hourly resolution) and the date when an activity between the two antenna a and b is registered, ID_a and ID_b are the identifiers of the transmitting and receiving antenna respectively, *NumCalls* is the number of calls started from a and received by b in the time interval and *TotTime* is the sum of the durations (in seconds) of all the calls that took place in the specific time interval. An example of the entry of the dataset is the following:

$$\langle 2011-12-05/00:00:00, 15, 913, 10, 3008 \rangle$$

which means that in the date 2011-12-05/00:00:00 there were registered 10 different calls, with a total duration of 3008 seconds, transmitted from the antenna with ID 15 and received by the antenna with ID 913.

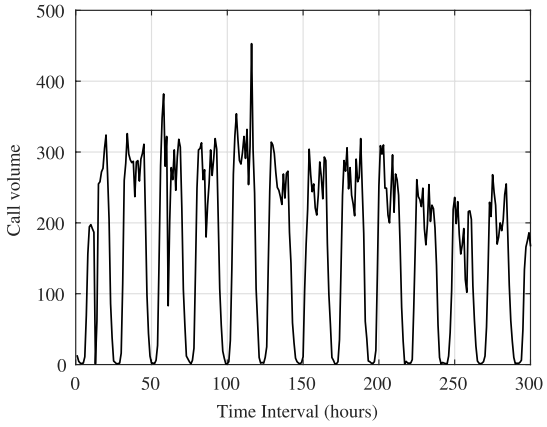


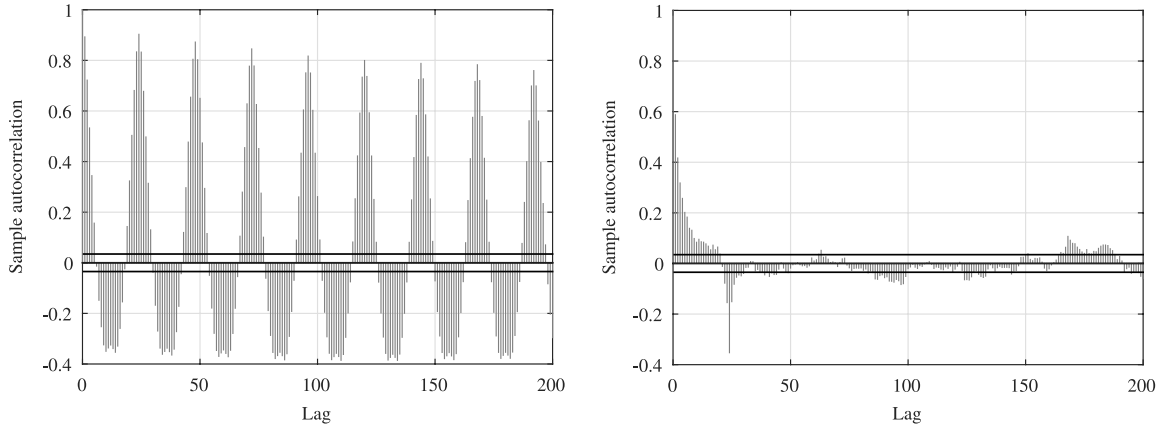
Fig. 1. The load profile of `ts2` for the first 300 time intervals.

In the following, we explain how we preprocessed the data and how the TSs have been generated. We selected a specific antenna and we retrieved from the dataset all the records relative to the telephone activity where the specific antenna is involved. We generated the following 7 TSs:

- `ts1`: constant input (a TS with all values set to 1). This is a standard practice in prediction with neural architectures, since the constant input acts as a bias for the individual neurons of the network (Jaeger, 2002).
- `ts2`: number of incoming calls in the area covered by the antenna.
- `ts3`: volume in minutes of the incoming calls in the area covered by the antenna.
- `ts4`: number of outgoing calls in the area covered by the antenna.
- `ts5`: volume in minutes of the outgoing calls in the area covered by the antenna.
- `ts6`: hour of the day when the telephone activity has been registered.
- `ts7`: day of the week when the telephone activity has been registered.

In each TS there is a (small) number of missing values because, if in a given hour it was not registered any outgoing and/or incoming telephone activity involving the considered antenna, the relative entries do not appear in the database. Since we require all the TSs to have the same length and to have a value in each time interval from the observation period, before building the TSs we inserted an entry with a value "0" in the dataset in order to fill the missing entries, e.g. $\langle 2011-12-17/4:00:00, 13, 67, 0, 0 \rangle$. Another issue concerns the presence of corrupted data, which are marked by a "−1" in the dataset, and they represent periods when the telephone activity was not registered correctly. In this case, we have used the technique described in Shen and Huang (2005); the missing values were replaced with the average value of the corresponding periods (i.e., same weekday and hour of the day) in the two adjacent weeks. In Fig. 1 we report the profile of `ts2` relatively to the load in the first 300 time intervals, being 1 h the length of each time interval.

Before processing the TS, a common procedure consists in applying some kind of normalization to the data like standardization



(a) Autocorrelation. (b) Autocorrelation after seasonal differencing.

Fig. 2. In (a) is drawn the autocorrelation of `ts2`. As we can notice there is a strong correlation component at every 24 lags. In (b) the autocorrelation of the same TS after a seasonal differencing. We can notice that TS now is deseasonalized and there are no other seasonalities.

or rescaling, which is successively reversed when the forecast value must be returned. Transforming the data with a non-linear function like square-root, logarithm or hyperbolic tangent (Andrews & Cunningham, 1995; Ghiassi et al., 2005; Ibrahim & L'Ecuyer, 2013; Shen & Huang, 2008; Weinberg, Brown, & Stroud, 2007) can be done for stabilizing the variance in the data, without altering their underlying structure. A log-transformation allows also to capture a multiplicative seasonal pattern by models which are inherently additive. Then, in presence of long term trends or when the residuals show a marked increment in variance over time and the amplitudes of the seasonal cycles should be normalized, a non-linear transformation should be used.

A preemptive analysis of the TS can help us to understand the nature of the seasonality and the variance in the data. In this preprocessing phase it is also important to identify the presence of multiple seasonality, hidden daily and/or hourly patterns, which could be better treated using multiple forecasting models (Deihimi & Showkati, 2012; Niu et al., 2012) or with models designed for dealing with multiple seasonality (Taylor, 2003, 2010, 2012). Fourier frequency analysis and the autocorrelation function up to sufficient high number of lags are the most commonly used tools for seasonality study. As expected, in the autocorrelation plot of `ts2` in Fig. 2(a), we can notice that the TS shows an obvious seasonality pattern every 24 h and it contains a trend which makes the seasonality slightly additive but not multiplicative. Furthermore, we can exclude the presence of a second, less obvious seasonality by looking at the autocorrelation plot of the TS after a seasonal differencing (Franses, 1991) in Fig. 2(b). The same considerations hold also for the remaining TSs, that we do not report here for the sake of conciseness.

For identifying the presence of weekly recurrent patterns we consider the average daily load in the TSs relatively to the different days of the week. As an example we report in Fig. 3 the average daily load profile of `ts2` in the 7 days of the week: as we can see, the profile is very similar in the different days of the week and the call volumes always show predictable, repetitive patterns. For example, there is typically a peak around 9:00, followed by two other peaks at 15:00 and 19:00. For what concerns the variance, our data appear to possess some heteroscedasticity (i.e., nonconstant variance) but not overdispersion (i.e., variance greater than the mean), as can be seen in Fig. 4 which depicts the standard deviation computed over the call loads in different days of the week. Because of the absence of the overdispersion, there is not a particular need of stabilizing the variance. Also, since there are no multiplicative elements in the seasonality, it is sufficient to normalize the values of the data using a rescaling or a standardization. In particular, we choose to rescale the values of the TSs in the interval [0, 1]. Finally, since we do not observe the presence of multiple seasonality in the data or well defined weekly patterns, a single forecasting model can be used in our STFL task. As a final remark we want to underline that in the analyzed TSs we consider *every* day in the observed period, without doing any a-priori filtering of the anomalous days, like holidays or days with an anomalous number of calls, which are commonly discarded in the preprocessing step or modeled as separate variables (Andrews & Cunningham, 1995; Ibrahim & L'Ecuyer, 2013; Shen & Huang, 2008).

4. The proposed forecasting method

The STFL problem consists in predicting the next l values of a TS given the current observations, being l a short forecast horizon. A common case in many applications is the 1 and 24 h ahead predictions, which in our case, where the time resolution is 1 h, consist in forecasting the next 1 and 24 values of the TS using the information gathered up to the present time. The system that we propose uses an ESN for forecasting the future values of the TS. In the following, we discuss the general structure of an ESN, the different methodologies used for training its readout and the procedure for optimizing the configuration parameters of the network and for selecting the set of exogenous variables to be considered in the forecasting procedure.

4.1. Structure of a echo state network

A schematic depiction of an ESN with a single output is shown in Fig. 5. It is composed by three components, namely the input layer, a recurrent reservoir, and a readout. Accordingly, the current output of an ESN is computed in two distinct phases. First, the N_i-dimensional input vector $\mathbf{x}[n] \in \mathbb{R}^{N_i}$ is given as input to the recurrent reservoir, whose internal state $\mathbf{h}[n-1] \in \mathbb{R}^{N_r}$ is updated according to the state equation:

$$\mathbf{h}[n] = f_{\text{res}}(\mathbf{W}_i^r \mathbf{x}[n] + \mathbf{W}_r^r \mathbf{h}[n-1] + \mathbf{w}_o^r y[n-1]), \tag{1}$$

where $\mathbf{W}_i^r \in \mathbb{R}^{N_r \times N_i}$, $\mathbf{W}_r^r \in \mathbb{R}^{N_r \times N_r}$ and $\mathbf{w}_o^r \in \mathbb{R}^{N_r}$ are set at the beginning of the learning process, and they remain unaltered afterward. $f_{\text{res}}(\cdot)$ in Eq. (1) is a suitable non-linear function, typically of sigmoid shape, and $y[n-1] \in \mathbb{R}$ is the previous scalar output of the network. In our case, we have $f_{\text{res}}(\cdot) = \tanh(\cdot)$. In the second phase, the ESN's prediction is computed according to:

$$y[n] = \left(\mathbf{w}_i^o\right)^T \mathbf{x}[n] + \left(\mathbf{w}_r^o\right)^T \mathbf{h}[n], \tag{2}$$

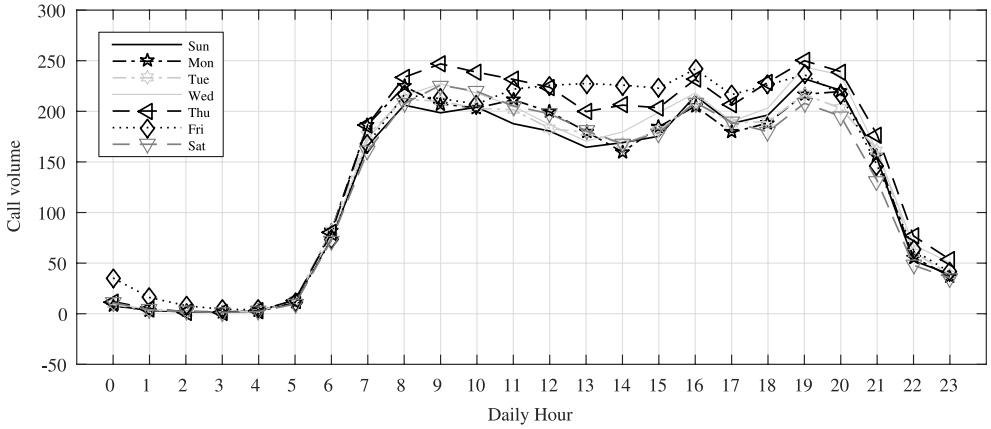


Fig. 3. The average load values in the different days of the week. The load profile is quite similar in each day and we can observe 3 different peaks at hours 9:00, 15:00 and 19:00 in each curve.

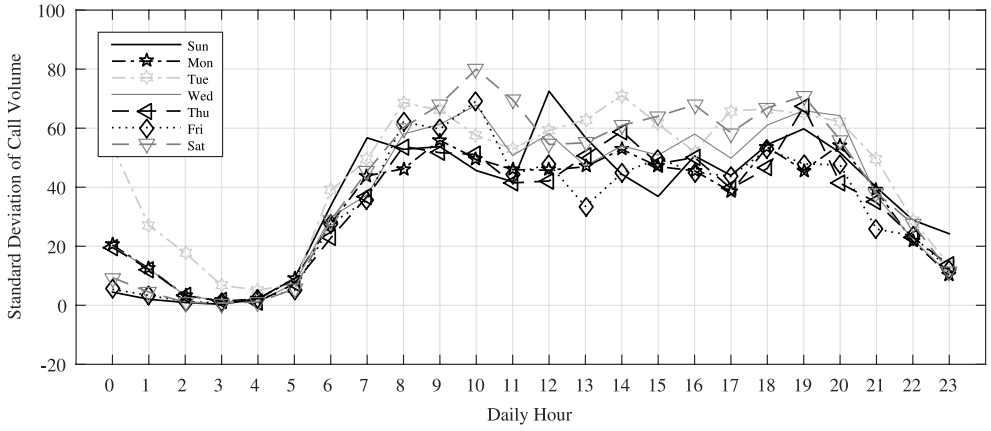


Fig. 4. The variance on of the loads in the different days is not constant, however the values of the variance are not greater than the mean.

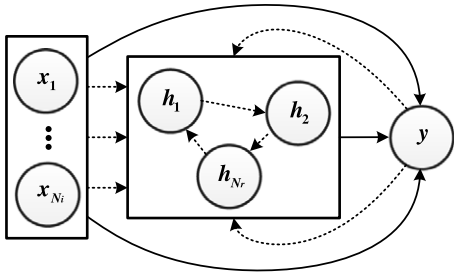


Fig. 5. Schematic depiction of an ESN. Random connections are shown with dashed lines, while trainable connections are shown with solid lines.

where $\mathbf{w}_i^o \in \mathbb{R}^{N_i}, \mathbf{w}_r^o \in \mathbb{R}^{N_r}$ are trainable connections. It is possible to add an additional (invertible) non-linearity in Eq. (2), although it was not considered in this paper. The difference between fixed and adaptable weights is shown in Fig. 5 with the use of fixed and dashed lines, respectively. Additionally, to increase the overall stability, it is possible to insert a small uniform noise term to the state update in Eq. (1), before computing the non-linear transformation $f_{\text{res}}(\cdot)$ (Jaeger, 2002).

Finally, a few words should be spent on the choice of the matrix $\mathbf{W}_r^r$. According to the ESN theory, the reservoir must satisfy the so-called 'echo state property' (ESP) (Lukoševičius & Jaeger, 2009). This means that the effect of a given time-instant on the state of the reservoir must vanish in a finite number of time-instants. A widely used rule-of-thumb is to rescale the matrix $\mathbf{W}_r^r$ to have $\rho(\mathbf{W}_r^r) < 1$, where $\rho(\cdot)$ denotes the spectral radius operator. For simplicity, we use this strategy in this paper, and we refer the interested reader to (Yildiz, Jaeger, & Kiebel, 2012) for recent theoretical studies on this subject. If the ESP is satisfied, an ESN with a suitably large N_r can approximate any non-linear filter with bounded memory to any given level of accuracy (Lukoševičius & Jaeger, 2009).

4.2. Readout training methods

To train the ESN, let us consider a sequence of Q desired input–outputs pairs given by:

$$(\mathbf{x}[1], d[1]) \ldots, (\mathbf{x}[Q], d[Q]). \tag{3}$$

In our case, the input vector $\mathbf{x}[i]$ is a 7-dimensional vector containing the values of the 7 TSs described in Section 3 at the ith time interval, while the output is given by:

$$d[i] = x_k[i + l], \tag{4}$$

where l and k are set *a priori* and define the forecasting horizon and the desired TS to be predicted. In the initial phase of training, called 'warming', the inputs are fed to the reservoir in accordance with Eq. (1), resulting in a sequence of internal states $\mathbf{h}[1], \ldots, \mathbf{h}[Q]$. Since the outputs of the ESN are not available for feedback, the desired output is used instead in Eq. (2) (so-called 'teacher forcing'). The results of this operation are stacked in the state matrix $\mathbf{H} \in \mathbb{R}^{Q \times N_i + N_r}$ and output vector $\mathbf{d} \in \mathbb{R}^Q$ as:

$$\mathbf{H} = \begin{bmatrix} \mathbf{x}^T[1]\ \mathbf{h}^T[1] \\ \vdots \\ \mathbf{x}^T[Q]\ \mathbf{h}^T[Q] \end{bmatrix}, \tag{5}$$

$$\mathbf{d} = \begin{bmatrix} d[1] \\ \vdots \\ d[Q] \end{bmatrix}. \tag{6}$$

Practically, it is possible to discard the initial D rows from Eqs. (5) and (6), since they refer to a transient phase in the ESN's behavior. We refer to them as the *wash-out* elements. It is also possible to handle multiple sequences in input, but it is not the concern of this work.

Clearly, at this point the resulting training problem is a standard linear regression, which can be solved in a large variety of ways. In our work, we compared 4 different algorithms to this end, namely the least-square regression, the elastic net penalty, and the ν-SVR with a linear kernel function and a Gaussian kernel function. These are briefly summarized in the following.

Least-square regression. The least-square regression (LSR) is the algorithm originally used for training the readout proposed by Jaeger (2001). It consists in the following regularized least-square problem:

$$\mathbf{w}_{\text{ls}}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^{N_i+N_r}} \frac{1}{2}\|\mathbf{H}\mathbf{w}-\mathbf{d}\|_2^2 + \frac{\lambda}{2}\|\mathbf{w}\|_2^2, \tag{7}$$

where $\mathbf{w}_{\text{ls}} = \left[\mathbf{w}_i^o\ \mathbf{w}_r^o\right]^T$ and $\lambda \in \mathbb{R}^+$ is a positive scalar known as *regularization factor*. A solution of problem (7) can be obtained in closed form as:

$$\mathbf{w}_{\text{ls}}^* = \left(\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I}\right)^{-1}\mathbf{H}^T\mathbf{d}. \tag{8}$$

Whenever $N_r + N_i > Q$, Eq. (8) can be computed more efficiently by rewriting it as:

$$\mathbf{w}_{\text{ls}}^* = \mathbf{H}^T\left(\mathbf{H}\mathbf{H}^T + \lambda\mathbf{I}\right)^{-1}\mathbf{d}. \tag{9}$$

Elastic net penalty. A second alternative is the use of the elastic net penalty (ENP) (Zou & Hastie, 2005). In this case, we consider the introduction of an additional L_1 penalty in Eq. (7), resulting in:

$$\mathbf{w}_{\text{el}}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^{N_i+N_r}} \frac{1}{2}\|\mathbf{H}\mathbf{w}-\mathbf{d}\|_2^2 + \lambda\left\{\frac{(1-\alpha)}{2}\|\mathbf{w}\|_2^2 + \alpha\|\mathbf{w}\|_1\right\}, \tag{10}$$

where $\|\mathbf{w}\|_1 = \sum_i |w_i|$, and $\alpha \in [0, 1]$ weight the relative influence of the L_2 and L_1 regularization terms. For $\alpha = 1$, Eq. (10) results in the LASSO algorithm (Tibshirani, 1996), which was investigated as a training algorithm for ESNs in Dutoit et al. (2009). While Eq. (10) requires a more sophisticated solver with respect to the standard least-square regression (Zou & Hastie, 2005), the inclusion of the L_1 penalty provides a sparse output vector, which in turn can be used as a pruning criterion for the reservoir (Dutoit et al., 2009; Scardapane, Nocco, Comminiello, Scarpiniti, & Uncini, 2014).

Linear SVR. A third approach that we investigate is the use of an SVM algorithm in the readout. This was originally proposed in Shi and Han (2007) (see also Li et al., 2012), where it was shown to be particularly robust in the case of chaotic time series. In particular, we consider the linear ν-SVR (LSVR) introduced in Schölkopf, Smola, Williamson, and Bartlett (2000), which is given by the following optimization problem:

$$\mathbf{w}_{\text{lsvr}}^* = \begin{cases} \min\limits_{\mathbf{w}\in\mathbb{R}^{N_i+N_r}} & \frac{1}{2}\|\mathbf{H}\mathbf{w}-\mathbf{d}\|_2^2 \\ & \quad + \lambda\left(\nu\epsilon + \frac{1}{Q}\sum_{i=1}^{Q}\left(\zeta_i + \zeta_i^*\right)\right) \\ \text{subject to} & \mathbf{w}^T\mathbf{s}[i] - d[i] \le \epsilon + \zeta_i, \\ & d[i] - \mathbf{w}^T\mathbf{s}[i] \le \epsilon + \zeta_i^*, \\ & \zeta_i, \zeta_i^* \ge 0 \end{cases} \tag{11}$$

where $\mathbf{s}[i] = \left[\mathbf{x}^T[i]\ \mathbf{h}^T[i]\right]^T$. In Eq. (11), each error greater than ϵ is penalized linearly, while errors less than ϵ are not penalized. The term ϵ is minimized by the inclusion of an additional regularization term weighted by a scalar $\nu > 0$. This formulation has a strong theoretical justification in the statistical learning theory framework (Schölkopf et al., 2000). It is possible to show that the resulting hyperplane only depends on a subset of all the training data, which are denoted as support vectors. The parameter ν provides a lower bound on the number of desired support vectors.

Non-linear SVR. Finally, we consider a non-linear extension of the ν-SVR (NLSVR) with the use of a kernel function. In this case, the state $\mathbf{s}[i]$ is projected to an higher dimensional feature space $\phi(\mathbf{s}[i])$, and the ν-SVR is applied on the resulting space. Substituting this mapping in Eq. (11), it can be shown that the dual optimization problem can be written as:

$$\mathbf{w}_{\text{nlsvr}}^* = \begin{cases} \min\limits_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*\in\mathbb{R}^Q} & \frac{1}{2}\left(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*\right)\mathbf{K}\left(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*\right) + \mathbf{d}^T\left(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*\right) \\ \text{subject to} & \mathbf{1}^T\left(\boldsymbol{\alpha}-\boldsymbol{\alpha}^*\right) = 0, \\ & \mathbf{1}^T\left(\boldsymbol{\alpha}+\boldsymbol{\alpha}^*\right) \le \lambda\nu, \\ & 0 \le \alpha_i, \quad \alpha_i^* \le \frac{\lambda}{Q},\ i = \ldots, Q \end{cases} \tag{12}$$

where each entry K_{ij} is given by $\mathcal{K}(\mathbf{s}[i], \mathbf{s}[j])$, with $\mathcal{K}(\cdot,\cdot)$ being the reproducing kernel associated to the feature mapping. By an extension of the representer's theorem, the output of the ESN at a generic time-instant n in this case is given by:

$$y[n] = \sum_{i=1}^{Q}\left(\alpha_i - \alpha_i^*\right)\mathcal{K}(\mathbf{s}[i], \mathbf{s}[n]), \tag{13}$$

where α_i and α_i^* are the entries of the optimal solution to problem (12), and they are non-zero only for patterns that are support vectors. This is computationally more expensive than the previous solutions, but it provides a non-linear model in the original space, which is more flexible. With the linear kernel $\mathcal{K}(\mathbf{s}[i], \mathbf{s}[j]) = \mathbf{s}^T[i]\mathbf{s}[j]$ we recover the linear ν-SVR. For the non-linear version, we consider the Gaussian kernel given by:

$$\mathcal{K}(\mathbf{s}[i], \mathbf{s}[j]) = \exp\left\{-\frac{\|\mathbf{s}[i]-\mathbf{s}[j]\|_2^2}{2\sigma^2}\right\}, \tag{14}$$

where σ is denoted as the scale parameter.

4.3. Parameter optimization

The initial configuration of the ESN depends on a set of parameters which need to be properly tuned in order to maximize the performances achieved on the specific task at hand. A possible approach consists in using a genetic algorithm (GA), in order to identify an optimal parameter configuration. Since the variables that need to be tuned assume both integer and real values, we adopted a mix-integer GA based on MI-LXPM algorithm (Deep, Singh, Kansal, & Mohan, 2009), which optimizes a genetic code composed of real and integer values, defined in a suitable interval. The algorithm follows the standard procedure of genetic algorithms, which begin by creating a random initial population. Then, at each step the algorithm generates a new population using some of the individuals with high fitness in the current generation, called parents, which are chosen using a tournament selection with tournament size equal to 2. Individuals of the next generation are created either by making random changes to a single parent with a *Gaussian mutation*, which adds a random number taken from a Gaussian distribution with mean 0 to each entry of the parent vector, or by combining the vector entries of a pair of parents with a *Laplacian crossover* (Deep et al., 2009), with crossover fraction μ.

The algorithm also implements a form of elitism, importing the E individuals with the highest fitness in the next generation, where $E = \lceil \rho P \rceil$, being ρ the elitism rate, P the population size and $\lceil \cdot \rceil$ the floor operator.

It must be pointed out that the GA is a fundamentally heuristic optimization procedure, adopted to perform a pseudo-random search in the solution space when the objective function cannot be expressed in a closed form. Even if the GA does not guarantee the convergence to a global optimum, a number of conditions are used for triggering the stop criteria when a sufficiently good solution is identified. In particular, the optimization stops when: (i) a maximum number of generations G is reached; (ii) the average relative change in the fitness function value over a number of consecutive generations is below a given threshold τ_{stall}; (iii) the value of the fitness function for the best point in the current population is less than or equal to the fitness limit — usually 0 (or 1), if the fitness score is normalized and it must be minimized (or maximized).

Previous uses of GAs in the context of parameter optimization for ESNs (Deihimi & Showkati, 2012; Jiang, Berry, & Schoenauer, 2008; Xu, Lan, & Principe, 2005) have shown that its performance is comparable to an exhaustive search in most cases, and that the ESN itself is robust to small changes in its parameters. We took into account the recommendations reported in previous works, e.g. Venayagamoorthy and Shishir (2009), as guidelines to determine the search space of the genetic code. We use the GA also for selecting the subset of TSs to be considered as exogenous input variables and for setting the parameters of the specific algorithm that is used for training the readout of the network. In the following we define the search-spaces considered by the GA for each parameter, and the optional resolution of the search.

- For the size of the reservoir we selected the bounds [100, 1000], with a resolution of 100.
- The entries of the reservoir matrix $\mathbf{W}_r^r$ are extracted from an uniform distribution in $[-1, +1]$. Then, a given percentage p of values are set to 0, and the matrix is rescaled to obtain a desired spectral radius ρ^*. In the GA, ρ^* is optimized in the interval [0.5, 0.99] and the percentage p of the connectivity in the reservoir in [0.1, 0, 4].
- We add a small noise term in Eq. (1), extracted from a Gaussian distribution with zero mean and variance β^2. The optimal variance is selected in the interval [0.00001, 0.001].
- To provide additional flexibility, the input signal, the desired response, and the feedback signal are all scaled by three constant factors denoted by γ_{in}, γ_{out}, and γ_{feed} respectively. The first two are optimized in the interval [0.1, 1], while the third in the interval [0, 1]. The particular case $\gamma_{\text{feed}} = 0$ corresponds to an ESN without feedback.
- For the selection of the exogenous variables we used a set of Boolean values, which can be represented with the interval [0, 1] and a search resolution of 1.
- The search of the parameter λ used in all the four training algorithms takes place in the exponential sequence $\{2^c\}$, where the variable c is searched by the GA in the interval $[-10, 10]$ with resolution 1.
- The variable α used in ENP is searched in [0.00001, 1].
- The parameter ν used in both LSVR and NLSVR is searched in [0, 1].
- The parameters ϵ and σ are both optimized in $\{10^c\}$, where c is again an integer, searched this time in $[-5, 5]$.

The fitness of each genetic code is computed as the forecasting error on a validation set $\mathcal{S}_{\text{vs}}$, which contains the values that come after the ones of the training set in the original TS, with a suitable error function. In particular we used the Normalized Root Mean Squared Error (NRMSE) function (De Gooijer & Hyndman, 2006), defined as:

$$\text{NRMSE}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}}{y_{\max} - y_{\min}} \tag{15}$$

where $\hat{\mathbf{y}}$ is a vector of n predictions and $\mathbf{y}$ is the vector of true values, after removing the D wash-out elements. Additionally, $y_{\max}$ and $y_{\min}$ denote the maximum and minimum values of the vector $\mathbf{y}$.

5. Experiments and results

For testing our system we arbitrarily decided to forecast the values of the TS `ts2` which represents the number of incoming calls in the area covered by the antenna, using the remaining 6 TSs as exogenous variables. We observed similar performance for forecasting the number of outgoing calls. In the following we evaluate the results obtained on two different STFL problems, with leading times of 1-h and 24-h ahead respectively.

5.1. Configuration for the tests

The code was fully implemented in MATLAB. For what concerns the ESN we used a modified implementation of the Simple ESN toolbox (Jaeger, 2009), the prediction with ARIMA and ARIMAX were done using the relative software in the MATLAB financial toolbox. For the TES method instead, we implemented it according to the Holt–Winter's exponential smoothing algorithm. Each TS contains 3336 values, which represent the hourly measurements of different quantities in each TS. We divided each TS into 3 different sets, namely the training set $\mathcal{S}_{\text{tr}}$, the validation set $\mathcal{S}_{\text{vs}}$ and the test set $\mathcal{S}_{\text{ts}}$. We used the first 80% of the TS as $\mathcal{S}_{\text{tr}}$, of the remaining part the first 10% become the $\mathcal{S}_{\text{vs}}$ and the last 10% the $\mathcal{S}_{\text{ts}}$.

For the prediction with ARIMA and ARIMAX we used different models for each TS, that have been designed according to an initial analysis of the TSs, which is partially reported in Section 3. We refer to an ARIMA model using the standard notation ARIMA$(\mathbf{p}, \mathbf{d}, \mathbf{q}) \times (\mathbf{P}, \mathbf{D}, \mathbf{Q})_s$, where the term $(\mathbf{p}, \mathbf{d}, \mathbf{q})$ gives the order of the nonseasonal part, $(\mathbf{P}, \mathbf{D}, \mathbf{Q})$ the order of the seasonal part, being s the value of the seasonality (Ho et al., 2002). The adopted models for the TSs `ts1`-7 are the following:

- ARIMA$(0, 0, 0)$ for `ts1`
- ARIMA$(0, 1, 2) \times (0, 1, 1)_{24}$ for `ts2`
- ARIMA$(0, 1, 1) \times (0, 1, 1)_{24}$ for `ts3`
- ARIMA$(0, 1, 2) \times (0, 1, 1)_{24}$ for `ts4`
- ARIMA$(0, 1, 1) \times (0, 1, 24)_{24}$ for `ts5`
- ARIMA$(0, 0, 0) \times (0, 1, 0)_{24}$ for `ts6`
- ARIMA$(0, 1, 0) \times (1, 1, 1)_{24}$ for `ts7`.

For the ARIMA, ARIMAX and TES models the parameters of the models are evaluated using the values in $\mathcal{S}_{\text{tr}}$ and $\mathcal{S}_{\text{vs}}$. For the TES models, the optimal set of the parameters is obtained using a nonlinear least-squares optimization configured with the Levenberg–Marquardt algorithm. The optimal set of TSs to be used as exogenous inputs in the ARIMAX model was identified through an exhaustive search: we considered all the possible input configurations (2^6 in our case) and we chose as exogenous input variables the set of TSs which minimized the final forecasting error on $\mathcal{S}_{\text{vs}}$, evaluated with the error function in Eq. (15).

For training the ESN we optimized the network parameters using the GA, as described in Section 4.3, configured with a crossover rate $\mu = 0.8$, a population size $P = 50$, a maximum number of generations $G = 100$, a stop threshold $\tau_{\text{stall}} = 0.01$ and an elitism rate $\rho = 0.05$. During the optimization step (see Section 4.3), $\mathcal{S}_{\text{tr}}$ is used for training the ESN and the prediction

Table 1
Prediction results, expressed with NRMSE, obtained using TES, ARIMA, ARIMAX and ESN trained with 4 different approaches: ESN + LR is an ESN with the readout trained with linear regression, with ESN + ENP we refer to a network whose readout is trained with the elastic net penalty and with ESN + LSVR and ESN + NLSVR we refer to the training performed with SVR using a linear and a radial basis function kernel respectively. We also report for ARIMAX and ESN which TSs (`ts1,ts3-7`) are selected as exogenous variables for predicting the values of `ts2`. Best results for each task are highlighted in bold.

	Method	NRMSE	Ex vars
1 h	TES	0.2784	–
	ARIMA	0.2772	–
	ARIMAX	0.2769	4, 6
	ESN + LR	0.2192 ± 0.0043	4, 6
	ESN + ENP	0.2560 ± 0.0172	3, 4, 6
	ESN + LSVR	0.2222 ± 0.0034	4, 6
	ESN + NLSVR	**0.2111 ± 0.0011**	4, 6
24 h	TES	**0.3409**	–
	ARIMA	0.5163	–
	ARIMAX	0.4873	6, 7
	ESN + LR	0.3666 ± 0.0489	3, 6
	ESN + ENP	0.3843 ± 0.0635	4, 6
	ESN + LSVR	0.4106 ± 0.0771	3, 6, 7
	ESN + NLSVR	0.3544 ± 0.0017	3, 4, 6

accuracy is evaluated on $\mathcal{S}_{vs}$ using the NRMSE function of Eq. (15), which represents the reciprocal of the fitness for each genetic code. Once the optimization step is concluded in each forecasting model, a final prediction accuracy is evaluated on $\mathcal{S}_{ts}$, which quantifies the generalization capability of each system.

5.2. Results

In this section we report the results of the 1-h and 24-h ahead forecasts obtained on the $\mathcal{S}_{ts}$ with the different prediction models. We also report the optimal set of the exogenous variables identified by ESN and ARIMAX. For the ESN we report the results obtained using different training methods of the readout (see Section 4.2). Because of the stochastic nature of the ESN, we repeated the optimization, the training of the network and the final test on $\mathcal{S}_{ts}$ 10 different times: what we report is the average value of the prediction error obtained and the TSs that have been selected more frequently as exogenous variables. Additionally, because of the initial transient phase of the ESN, we discard the first 50 output values as wash-out elements in order to allow the system to reach a steady behavior. In Fig. 6 we plotted the predicted values of the last 100 elements of `ts2` using ARIMAX, TES and the ESN trained with the SVR configured with a radial basis function kernel. In Table 1 we report the complete results and the optimal set of exogenous variables selected.

As we can see from Table 1, the higher prediction accuracy with a leading time of 1 h is achieved by ESN configured with different training methods for the readout. In particular, the best result is obtained by ESN using SVR with the Gaussian kernel (ESN + NLSVR). For what concerns the 24-h ahead prediction, even if the NRMSE is low for the ESN-based forecast system using the linear regression (ESN + LR) and ESN + NLSVR, which also in this case obtains better performances with respect to other training methods, the best performances are achieved by TES. This outcome is expected, since ESN has proven to be a very effective tool in prediction with short leading time, while the performance of TES on forecasts with long leading times are known to be better than more complex models (Taylor, 2008), especially if the TS to be predicted is very regular, like in our case.

The poorest results are returned by the ESN trained with the elastic net penalty algorithm (ESN + ENP), which demonstrated to be not suitable for the considered TSF problem. Since the values that we report are the means of the error returned in 10 different runs we also report the variance which, as we can see from the table, is very small, especially in the results of ESN + LR, ESN + LSVR and ESN + NLSVR. This shows that, despite the random internal structure of the network and of the genetic optimization, the results are stable and the values obtained are significant. ARIMA and ARIMAX achieved similar, acceptable results, but they are not able to outperform ESN in the short time forecasting or TES in the 24-h ahead prediction.

Table 2
MI values between the TSs `ts2-7`. Note that `ts1` is a constant input and is not considered in the computation of the MI.

	`ts3`	`ts4`	`ts5`	`ts6`	`ts7`
`ts2`	1.003	1.669	0.737	0.589	0.013
`ts3`	–	0.867	0.895	0.521	0.014
`ts4`	–	–	0.881	0.594	0.019
`ts5`	–	–	–	0.529	0.018
`ts6`	–	–	–	–	0

In order to discuss the optimal subset of exogenous variables that have been selected by the different methods, we analyze the non-linear pairwise relations between the `ts2`, …, `ts7` (we do not consider `ts1` which is the constant input) by evaluating their Mutual Information (MI), which provides a general measurement of the non-linear dependence between two random variables. Due to the presence of continuous values in the TSs, we used the kernel based estimator described in Moon, Rajagopalan, and Lall (1995). The MI values are reported in Table 2, they are directly proportional to the degree of dependence between the TSs.

The TSs which are selected the most as external inputs for supporting the prediction, are `ts4` and `ts6`. As we can see from the MI values in Table 2, `ts4` – the TS containing the values relative to the number of outgoing calls in the area covered by the antenna – is the most related with `ts2` — the TS whose values we want to predict and that contains the number of incoming calls processed by the same antenna. The selection of `ts4` can be justified by the fact that it contains the same type of values of `ts2`, also it is reasonable to assume that in a given time interval the quantity of incoming and outgoing calls is related, since they are both referred to the period of activity of the individuals that are served by the considered antenna. For what concerns `ts6`, even if it appears to be weakly related with `ts2` according to their MI, the information it contains relative to the hour catches very well the seasonality of the TS. Since all the TSs that we consider have a strong repetitive pattern in the load profile during the day (see Section 3), considering the hour in the prediction is reasonable. On the other hand, in the study of the TSs we noticed that in different days of the week the load profile appears to be the same. For this reason, `ts7` is rarely considered as external variable since, due to the aforementioned regularity, knowing which is the day of the week is irrelevant in forecasting the load of the calls. For each method we measured the execution time, which is displayed in the bar graph in Fig. 7, where the height of each column is proportional to the number of seconds required for training the predictor and forecasting the values $\mathcal{S}_{ts}$. As we can see, the ESN which uses the linear regression or elastic net penalty as training algorithm for the readout is the fastest. In particular, the linear regression can be simply computed through the inversion of a matrix (see Section 4). The SVR algorithm instead, is a more complex procedure which requires a longer computational time. For what concerns the other predictors, the optimization of the parameters in the TES and the updating procedure of the model, due to its simplicity, are faster than ARIMA. As expected, ARIMAX requires the longest computational time, since the parameter estimation and the updating must be repeated for each of the considered TSs. Finally, Table 3 contains the values of the configuration parameters of the ESN identified by the GA, for each of the training algorithms

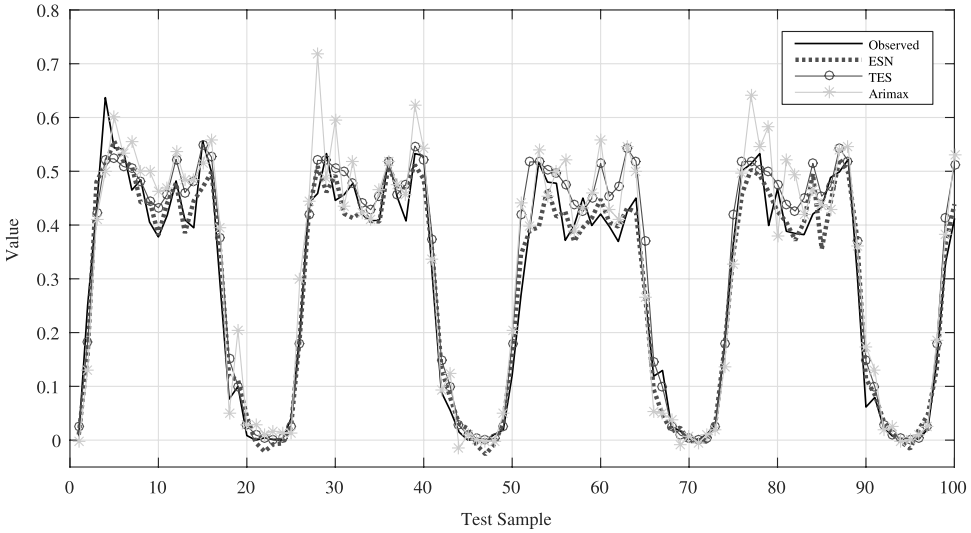


Fig. 6. Results of the prediction using TES, ARIMAX and ESN trained with SVR with a Gaussian kernel of the last 100 values of `ts2`, considering a forecast horizon of 1 h.

Table 3
Mean and standard deviation of the optimal configuration values of the ESN returned by the GA in 10 different optimizations, which are reservoir size (**rs**), spectral radius (ρ), regularization parameter (λ), ENP parameter (α), SVR parameter (ν), kernel parameter (σ).

		rs	ρ	λ	α	ν	σ
1 h	LR	683 ± 149	0.98 ± 0.01	0.04 ± 0.05	–	–	–
	ENP	886 ± 64	0.95 ± 0.04	0 ± 0	0.20 ± 0.40	–	–
	LSVR	732 ± 214	0.92 ± 0.05	8.40 ± 13.57	–	0.48 ± 0.12	4000 ± 5477
	NLSVR	728 ± 192	0.94 ± 0.03	320.80 ± 446.72	–	0.79 ± 0.11	0.01 ± 0
24 h	LR	747 ± 136	0.93 ± 0.03	0 ± 0	–	–	–
	ENP	742 ± 193	0.95 ± 0.06	0 ± 0	0.10 ± 0.12	–	–
	LSVR	682 ± 158	0.86 ± 0.10	19.50 ± 26.02	–	0.31 ± 0.10	$22\,000 \pm 43\,817$
	NLSVR	638 ± 206	0.66 ± 0.15	410.60 ± 427.18	–	0.38 ± 0.32	0.04 ± 0.05

that we considered. The number that we report are the average values and the standard deviations obtained in the 10 different runs for the two TSF problems, namely the 1-h and 24-h ahead predictions. As we can observe, in average the network requires a relatively large reservoir to solve the prediction task, as evidenced by the second column. Consequently, the network also requires a high degree of memory, as shown by the values close to unity assumed by the spectral radius ρ. Despite the standard deviation of the optimal reservoir size (**rs**) and values of the parameters λ and σ are very high, the achieved results are relatively stable. In fact, the standard deviation in NRMSE (reported in Table 1) is very small, even if the ESN is configured using parameters with very different values. This underline a low degree of sensitivity to the values of the parameters and the stability of the predictor model which uses the ESN, that behaves in a similar way with different configurations. A few additional considerations can be made by looking at Table 3. In particular, the low performance of ESN + ENP can be attributed to the small values assumed by α, meaning that encouraging sparsity does not seem profitable in this context. A similar behavior is observed for the SVR, as evidenced by the large values assumed by ν.

6. Conclusions and future works

In this work we have addressed the TSF problem relative to the prediction of the load of incoming calls in a cell of a mobile network, considering a set of TSs containing additional information on the telephone activity on the same cell. We considered two different prediction problems, which are the 1-h and 24-h ahead forecast, using an ESN trained with four different algorithms, two of which are novel in the context of ESN. Additionally, we compared the results obtained with standard prediction models,

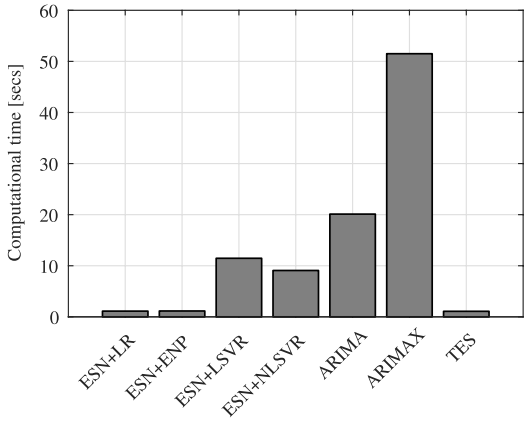


Fig. 7. Computational time in seconds required for training the system and forecasting the elements in $\mathcal{S}_{\text{ts}}$, using the considered prediction models.

namely ARIMA, ARIMAX and TES. We used a GA for tuning the parameters of the ESN and for selecting the set of additional TSs to be considered as exogenous variables in the forecast. The results obtained confirmed the high accuracy of ESN in the load prediction relative to short leading times. The best performances are achieved using the linear regression and SVR with Gaussian kernel for training the readout of the network. For longer forecast horizon instead, the TES seemed to be the most suitable system for the considered TSF problem, performing slightly better than ESN. During the training phase, the GA selected the TSs relative to the hour of the day and the load of outgoing calls as the most relevant exogenous variables; the result is in compliance with the nature of the data, confirming the effectiveness of our approach.

In a future work we plan to use ESN with more complex structures, more advanced training methods for the readout and reservoirs with additional properties, like the intrinsic plasticity (Steil, 2007) and the lateral inhibition (Xue, Yang, & Haykin, 2007). More in general, we plan to continue exploring the problem of forecasting the telephonic load in the cells of a mobile network, fundamental for an effective design of the network itself, and necessary for a reliable estimation of the employment of the resources of a telephone operator.

References

Aksin, Z., Armony, M., & Mehrotra, V. (2007). The modern call center: A multi-disciplinary perspective on operations management research. *Production and Operations Management*, *16*(6), 665–688.

Aldor-Noiman, S., Feigin, P. D., & Mandelbaum, A. (2009). Workload forecasting for a call center: Methodology and a case study. *The Annals of Applied Statistics*, *3*(4), 1403–1447.

Anderson, O.D. (1976). Time series analysis and forecasting: the Box–Jenkins approach. Butterworths L.

Andrews, B. H., & Cunningham, S. M. (1995). LL Bean improves call-center forecasting. *Interfaces*, *25*(6), 1–13.

Antipov, A., & Meade, N. (2002). Forecasting call frequency at a financial services call centre. *Journal of the Operational Research Society*, *53*(9), 953–960.

Blondel, V.D., Esch, M., Chan, C., Clérot, F., Deville, P., & Huens, E. et al. (2012). Data for development: the D4D challenge on mobile phone data.

Bunn, D. W. (2000). Forecasting loads and prices in competitive power markets. *Proceedings of the IEEE*, *88*(2).

Butcher, J. B., Verstraeten, D., Schrauwen, B., Day, C. R., & Haycock, P. W. (2013). Reservoir computing and extreme learning machines for non-linear time-series data analysis. *Neural Networks*, *38*, 76–89.

Deep, K., Singh, K. P., Kansal, M. L., & Mohan, C. (2009). A real coded genetic algorithm for solving integer and mixed integer optimization problems. *Applied Mathematics and Computation*, *212*(2), 505–518.

De Gooijer, J. G., & Hyndman, R. J. (2006). 25 years of time series forecasting. *International Journal of Forecasting*, *22*(3), 443–473.

Deihimi, A., & Showkati, H. (2012). Application of echo state networks in short-term electric load forecasting. *Energy*, *39*(1), 327–340.

Dutoit, X., Schrauwen, B., Van Campenhout, J., Stroobandt, D., Van Brussel, H., & Nuttin, M. (2009). Pruning and regularization in reservoir computing. *Neurocomputing*, *72*(7), 1534–1546.

Ferreira, A., Ludermir, T., de Aquino, R., Lira, M., & Neto, O. (2008). Investigating the use of reservoir computing for forecasting the hourly wind speed in short-term. In *2008 IEEE international joint conference on neural networks, IJCNN'08, June* (pp. 1649–1656).

Franses, P. H. (1991). Seasonality, non-stationarity and the forecasting of monthly time series. *International Journal of Forecasting*, *7*(2), 199–208.

Ghiassi, M., Saidane, H., & Zimbra, D. K. (2005). A dynamic artificial neural network model for forecasting time series events. *International Journal of Forecasting*, *21*(2), 341–362.

Ho, S. L., Xie, M., & Goh, T. N. (2002). A comparative study of neural network and Box–Jenkins ARIMA modeling in time series prediction. *Computers & Industrial Engineering*, *42*(2), 371–375.

Huang, C.-M., Huang, C.-J., & Wang, M.-L. (2005). A particle swarm optimization to identifying the ARMAX model for short-term load forecasting. *IEEE Transactions on Power Systems*, *20*(2), 1126–1133.

Ibrahim, R., & L'Ecuyer, P. (2013). Forecasting call center arrivals: Fixed-effects, mixed-effects, and bivariate models. *Manufacturing and Service Operations Management*, *15*(1), 72–85.

Ilies, I., Jaeger, H., Kosuchinas, O., Rincon, M., Sakenas, V., & Vaskevicius, N. (2007). *Stepping forward through echoes of the past: forecasting with echo state networks. Tech. rep*. Germany: Jacobs University Bremen.

Jaeger, H. (2001). *The echo state approach to analysing and training recurrent neural networks. Tech. rep., Technical report GMD report 148*. German National Research Center for Information Technology.

Jaeger, H. (2002). Adaptive nonlinear system identification with echo state networks. In *Advances in neural information processing systems* (pp. 593–600).

Jaeger, H. (2009). Simple and very simple Matlab toolbox for echo state networks. Avalaible online at: http://organic.elis.ugent.be/node/129.

Jaeger, H., & Haas, H. (2004). Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, *304*(5667), 78–80.

Jiang, F., Berry, H., & Schoenauer, M. (2008). Supervised and evolutionary learning of echo state networks. In *Parallel problem solving from nature–PPSN X* (pp. 215–224). Springer.

Kalekar, P.S. (2004). Time series forecasting using Holt–Winters exponential smoothing. Kanwal Rekhi School of Information Technology 4329008, 1–13.

Li, D., Han, M., & Wang, J. (2012). Chaotic time series prediction based on a novel robust echo state network. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(5), 787–799.

Lin, X., Yang, Z., & Song, Y. (2009). Short-term stock price prediction based on echo state networks. *Expert Systems with Applications*, *36*(3, Part 2), 7313–7317.

Lukoševičius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, *3*(3), 127–149.

Moon, Y.-I., Rajagopalan, B., & Lall, U. (1995). Estimation of mutual information using kernel density estimators. *Physical Review E*, *52*(3), 2318–2321.

Niu, D., Ji, L., Xing, M., & Wang, J. (2012). Multi-variable echo state network optimized by Bayesian regulation for daily peak load forecasting. *Journal of Networks*, *7*(11), 1790–1795.

Pearlmutter, B. A. (1995). Gradient calculations for dynamic recurrent neural networks: A survey. *IEEE Transactions on Neural Networks*, *6*(5), 1212–1228.

Peng, Y., Lei, M., Li, J.-B., & Peng, X.-Y. (2014). A novel hybridization of echo state networks and multiplicative seasonal ARIMA model for mobile communication traffic series forecasting. *Neural Computing and Applications*, *24*(3–4), 883–890.

Qingsong, S., Xiangmo, Z., Zuren, F., Yisheng, A., & Baohua, S. (2011). Hourly electric load forecasting algorithm based on echo state neural network. In *2011 Chinese control and decision conference, CCDC'11, May* (pp. 3893–3897).

Ruiz, P. A., & Gross, G. (2008). Short-term resource adequacy in electricity market design. *IEEE Transactions on Power Systems*, *23*(3), 916–926.

Sacchi, R., Ozturk, M., Principe, J., Carneiro, A., & da Silva, I. (2007). Water inflow forecasting using the echo state network: a Brazilian case study. In *2007 IEEE international joint conference on neural networks, IJCNN'07, August* (pp. 2403–2408).

Scardapane, S., Dianhui, W., & Panella, M. (2015). A decentralized training algorithm for echo state networks in distributed big data applications. *Neural Networks*, http://dx.doi.org/10.1016/j.neunet.2015.07.006. in press.

Scardapane, S., Nocco, G., Comminiello, D., Scarpiniti, M., & Uncini, A. (2014). An effective criterion for pruning reservoir's connections in echo state networks. In *2014 IEEE/INNS international joint conference on neural networks*, (IJCNN'14). (pp. 1205–1212). INNS/IEEE.

Schölkopf, B., Smola, A. J., Williamson, R. C., & Bartlett, P. L. (2000). New support vector algorithms. *Neural Computing*, *12*(5), 1207–1245.

Shen, H., & Huang, J. Z. (2005). Analysis of call centre arrival data using singular value decomposition. *Applied Stochastic Models in Business and Industry*, *21*(3), 251–263.

Shen, H., & Huang, J. Z. (2008). Interday forecasting and intraday updating of call center arrivals. *Manufacturing and Service Operations Management*, *10*(3), 391–410.

Shi, Z., & Han, M. (2007). Support vector echo-state machine for chaotic time-series prediction. *IEEE Transactions on Neural Networks*, *18*(2), 359–372.

Showkati, H., Hejazi, A., & Elyasi, S. (2010). Short term load forecasting using echo state networks. In *2010 IEEE international joint conference on neural networks, IJCNN'10, July* (pp. 1–5).

Soyer, R., & Tarimcilar, M. M. (2008). Modeling and analysis of call center arrival data: A Bayesian approach. *Management Science*, *54*(2), 266–278.

Steil, J. J. (2007). Online reservoir adaptation by intrinsic plasticity for backpropagation–decorrelation and echo state learning. *Neural Networks*, *20*(3), 353–364.

Taylor, J. W. (2003). Short-term electricity demand forecasting using double seasonal exponential smoothing. *Journal of the Operational Research Society*, *54*(8), 799–805.

Taylor, J. W. (2008). A comparison of univariate time series methods for forecasting intraday arrivals at a call center. *Management Science*, *54*(2), 253–265.

Taylor, J. W. (2010). Exponentially weighted methods for forecasting intraday time series with multiple seasonal cycles. *International Journal of Forecasting*, *26*(4), 627–646.

Taylor, J. W. (2012). Short-term load forecasting with exponentially weighted methods. *IEEE Transactions on Power Systems*, *27*(1), 458–464.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Varshney, S., & Verma, T. (2014). Half hourly electricity load prediction using echo state network. *International Journal of Science and Research*, *3*(6), 885–888.

Venayagamoorthy, G. K., & Shishir, B. (2009). Effects of spectral radius and settling time in the performance of echo state networks. *Neural Networks*, *22*(7), 861–863.

Verstraeten, D., Schrauwen, B., d'Haene, M., & Stroobandt, D. (2007). An experimental unification of reservoir computing methods. *Neural Networks*, *20*(3), 391–403.

Weinberg, J., Brown, L. D., & Stroud, J. R. (2007). Bayesian forecasting of an inhomogeneous Poisson process with applications to call center data. *Journal of the American Statistical Association*, *102*(480), 1185–1198.

Xu, D., Lan, J., & Principe, J. C. (2005). Direct adaptive control: an echo state network and genetic algorithm approach. In *2005 IEEE international joint conference on neural networks, 2005. IJCNN'05. Proceedings. Vol. 3* (pp. 1483–1486). IEEE.

Xue, Y., Yang, L., & Haykin, S. (2007). Decoupled echo state networks with lateral inhibition. *Neural Networks*, *20*(3), 365–376.

Yildiz, I. B., Jaeger, H., & Kiebel, S. J. (2012). Re-visiting the echo state property. *Neural Networks*, *35*, 1–9.

Zhang, G. P. (2001). An investigation of neural networks for linear time-series forecasting. *Computers & Operations Research*, *28*(12), 1183–1202.

Zhang, G. P., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks:: The state of the art. *International Journal of Forecasting*, *14*(1), 35–62.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, *67*(2), 301–320.