Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 200 (2022) 748–757

Procedia Computer Science

www.elsevier.com/locate/procedia

3rd International Conference on Industry 4.0 and Smart Manufacturing

Advanced statistical and machine learning methods for multi-step multivariate time series forecasting in predictive maintenance

Valentina Tessoni[a,b], Michele Amoretti[b]

[a] *Sidel, Via La Spezia 241a, Parma 43126, Italy*
[b] *University of Parma, Parco Area delle Scienze 181a, Parma 43124, Italy*

Abstract

The accurate prediction of failure events is of central interest to the field of predictive maintenance, where the role of forecasting is of paramount importance. In this paper, we present and compare some advanced statistical and machine learning methods for multi-step multivariate time series forecasting. Regarding statistical methods, we considered VAR, VMA, VARMA and Theta. The machine learning approaches we selected are variants of the Recurrent Neural Network model, namely ERNN, LSTM and GRU. All the considered methods have been evaluated in terms of accuracy, using 5 public datasets. As an extra contribution, we have implemented the multivariate version of the Theta method.

Keywords: predictive maintenance; multi-step multivariate time series forecasting; statistical methods; machine learning

1. Introduction

The accurate prediction of failure events is of central interest to the field of predictive maintenance [1], where the role of time series forecasting is of paramount importance. In particular, it is crucial to compute the remaining useful time (RUL), which is the useful life left on an asset at a particular time of operation [2, 3].

Montero Jimenez et al. [4] classify forecasting techniques in three groups: physics-based models, knowledge-based models, and data-driven models. Physics-based models require high skills on the underlying physics of the application. Knowledge-based models are based on cases or facts collected over the years of operation and maintenance. They are useful for diagnostics and provide explanatory results, but their performance on prognostics is more limited. In this

* Michele Amoretti. Tel.: +39-521-906390.
E-mail address: michele.amoretti@unipr.it

1877-0509
10.1016/j.procs.2022.01.273

sense, data-driven models are gaining popularity because of the improved availability of computational power and the production of Big Data.

In this paper, we present and compare some advanced statistical and machine learning methods for multi-step multivariate time series forecasting. Our work has been inspired by the M competitions,[1] whose ultimate purpose is to advance the theory of forecasting and improve its utilization by businesses and non-profit organizations. Their other goal is to compare the accuracy/uncertainty of machine learning and deep learning methods with standard statistical ones, and assess possible improvements versus the extra complexity and higher costs of using the various methods. Similarly, Yin et al. [5] conducted a systematic evaluation of forecasting methods to evaluate how their forecasting error depend on the features of the dataset and on the forecasting horizon.

Regarding statistical methods, we considered VAR, VMA, VARMA and Theta. The machine learning approaches we selected are variants of the Recurrent Neural Network model, namely ERNN, LSTM and GRU. All the considered methods have been evaluated in terms of accuracy, using 5 public datasets. As an extra contribution, we have implemented the multivariate version of the Theta method [6], starting from the bivariate one (which was the only available implementation, to the best of our knowledge).

The paper has the following structure. In Section 2, the concept of time series forecasting is presented in a more detailed fashion, with particular attention to accuracy measures. In Section 3, the considered statistical and machine learning methods are presented concisely. In Section 4, the experimental setup is illustrated. In Section 5, the experimental results are presented and discussed. Finally, in Section 6, some conclusions are drawn.

2. Time Series Forecasting

Time series forecasting uses the information in a time series to predict future values of that series. A *univariate* time series, as the name suggests, is a series with a single time-dependent variable. A *multivariate* time series, instead, has multiple time-dependent variables, each one depending not only on its past values but also on other variables. This dependency between variables is used for forecasting future values.

A time series forecasting problem that requires a prediction of multiple time steps into the future can be referred to as *multi-step* time series forecasting. Shorter time horizons are often easier to predict with higher confidence.

Many time series are characterized by trends and seasonal variations, which are relatively straightforward to identify. Serial correlation (also referred to as autocorrelation) measures the relationship between the current value of a variable and the values of the same variable from previous time periods. The study of serial correlations is commonly used in creating forecasting models.

2.1. Accuracy Measures

Before reviewing the most advanced forecasting methods, we present the accuracy measures that have been adopted in the M4 Competition [7]. The following notation is used. The actual value of the time series at point t is denoted as $\boldsymbol{y}^t$. The estimated forecast is denoted as $\hat{\boldsymbol{y}}^t$. The number of fitted points is n. The forecasting horizon is h. The seasonal period is m (e.g., 12 for monthly time series, 4 for quarterly, 24 for hourly). For non-seasonal time series (yearly, weekly and daily data) $m = 1$ s[7].

The symmetric mean absolute percentage error (sMAPE) is defined as

$$sMAPE = \frac{2}{h} \sum_{t=n+1}^{n+h} \frac{|\boldsymbol{y}^t - \hat{\boldsymbol{y}}^t|}{|\boldsymbol{y}^t| + |\hat{\boldsymbol{y}}^t|} * 100\% \tag{1}$$

[1] https://mofc.unic.ac.cy/history-of-competitions/

and the mean absolute scaled error (MASE) is defined as

$$MASE = \frac{1}{h} \frac{\sum_{t=n+1}^{n+h} |y^t - \hat{y}^t|}{\frac{1}{n-m} \sum_{t=m+1}^{n} |y^t - y^{t-m}|} \tag{2}$$

where $|\cdot|$ is the L1 norm. The overall weighted average (OWA) is the average of the Relative sMAPE and Relative MASE [7] using Naïve method as reference.

3. Forecasting Methods

A forecasting method is a predetermined sequence of steps that produces forecasts at future time periods [8]. Many forecasting methods have corresponding stochastic models that produce the same point forecasts and can also be used to generate prediction distributions and prediction intervals. A stochastic model makes assumptions about the process and the associated probability distributions.

The selection of a forecasting method depends on many factors: the context of the forecast, the relevance and availability of historical data, the degree of accuracy desirable, the time period to be forecast, the cost/ benefit (or value) of the forecast, and the time available for making the analysis [9].

3.1. Statistical Methods

In the following, we recap the major statistical methods for multivariate time series forecasting. We start from Naïve method, then we describe VAR, VMA e VARMA methods [10], and finally we illustrate the Theta method.

3.1.1. Naïve Method

Naïve forecasting is an estimating technique in which the current period's values are used as next period's forecast, without adjusting them or attempting to establish causal factors. This method is used only for comparison with the forecasts generated by more sophisticated techniques.

3.1.2. VAR, VMA and VARMA Methods

A vector auto regression process of order p, denoted as VAR(p) is a multivariate stochastic process $\boldsymbol{x}^t$ that fulfills the following equation:

$$\boldsymbol{x}^t = \boldsymbol{A}_1 \boldsymbol{x}^{(t-1)} + .. + \boldsymbol{A}_p \boldsymbol{x}^{(t-p)} + \boldsymbol{e}^t \tag{3}$$

where $\boldsymbol{e}^t$ is k-dimensional white noise and $\boldsymbol{A}_{1,\ldots,p}$ are $k \times k$ matrices.

Equation 3 is usually restated as

$$\boldsymbol{A}(\boldsymbol{L})\boldsymbol{x}^t = \boldsymbol{e}^t \tag{4}$$

in terms of the lag polynomial

$$\boldsymbol{A}(\boldsymbol{L}) = (\boldsymbol{I} - \boldsymbol{A}_1 \boldsymbol{L}_1 - .. - \boldsymbol{A}_p \boldsymbol{L}_p) \tag{5}$$

where the ith lag operator $\boldsymbol{L}_i$ is such that $\boldsymbol{L}_i \boldsymbol{y}^t = \boldsymbol{y}^{(t-i)}$.

A VAR process is stationary if it is stable, i.e., $\det \boldsymbol{A}(z) \neq 0$ for $|z| < 1$.

Similarly, a vector moving average process of order q, denoted as VMA(q) is a multivariate stochastic process $\boldsymbol{y}^t$ that fulfills the following equation:

$$\boldsymbol{y}^t = \boldsymbol{M}_0 \boldsymbol{z}^t + .. + \boldsymbol{M}_q \boldsymbol{z}^{(t-q)} \tag{6}$$

where $\boldsymbol{z}^t$ is a k-dimensional zero-mean white noise process. Equivalently,

$$\boldsymbol{y}^t = \boldsymbol{M}(\boldsymbol{L}) \boldsymbol{z}^t. \tag{7}$$

VAR and VMA processes can be combined to a so-called VARMA process, satisfying the following equation

$$\boldsymbol{y}^t = \boldsymbol{A}_1 \boldsymbol{y}^{(t-1)} + .. + \boldsymbol{A}_p \boldsymbol{y}^{(t-p)} + \boldsymbol{M}_0 \boldsymbol{z}^t + .. + \boldsymbol{M}_q \boldsymbol{z}^{(t-q)}. \tag{8}$$

Equivalently,

$$\boldsymbol{A}(\boldsymbol{L}) \boldsymbol{y}^t = \boldsymbol{M}(\boldsymbol{L}) \boldsymbol{z}^t. \tag{9}$$

The VARMA process is stationary if $\det \boldsymbol{A}(z) \neq 0$ for $|z| < 1$.

Forecasting a multivariate time series with VAR, VMA or VARMA requires, as a first step, to fit the selected method to the data. Once the parameters have been estimated, the method is applied to the time series whose future values must be forecast. In doing this, it is necessary to be aware that the resulting values are affected by the forecast error induced by the model and the forecast error induced by the estimation error.

3.1.3. Theta Method

The Theta method [11] is a univariate forecasting method based on the concept of modifying the second differences of a time series through the θ coefficient. When $\theta < 1$ the second differences are reduced, whereas when $\theta > 1$ the second differences are increased. This procedure produces new time series denoted as Theta-lines (or θ-lines), which are then extrapolated and combined to produce the forecast of the time series.

The Theta method was extended to the case of multivariate time series, in order to perform vector forecasting, by Thomakos and Nikolopoulos [6]. Consider a multivariate time series $\tilde{\boldsymbol{x}}^t$ of dimension k such that

$$\tilde{\boldsymbol{x}}^t = \boldsymbol{\mu} + \tilde{\boldsymbol{x}}^{(t-1)} + \boldsymbol{u}^t = \tilde{\boldsymbol{x}}^0 + \boldsymbol{\mu} t + \boldsymbol{S}^t \tag{10}$$

where $\mu \neq 0$ is the drift vector, the innovations $\boldsymbol{u}_t$ are assumed to follow a zero mean, stationary time series with finite second moments, and $\boldsymbol{S}_t = \sum_{j=1}^{t} \boldsymbol{u}_j$ is the stochastic trend of the cumulated innovations. Introducing $\boldsymbol{x}^t = \tilde{\boldsymbol{x}}^t - \tilde{\boldsymbol{x}}^0$, we obtain

$$\boldsymbol{x}^t = \boldsymbol{\mu} t + \boldsymbol{S}^t. \tag{11}$$

We define the multivariate θ-line to depend on a parameter matrix $\mathbf{\Theta}$ rather than a single parameter θ. We now have

$$\boldsymbol{Q}^t(\mathbf{\Theta}) = \mathbf{\Theta}\boldsymbol{x}^t + (\boldsymbol{I} - \mathbf{\Theta})\boldsymbol{\mu}t \tag{12}$$

where $\boldsymbol{I}$ is the k-dimensional identity matrix. Finally, we have the following forecast function:

$$\boldsymbol{F}^{(t+1)}(\mathbf{\Theta}) = \boldsymbol{\mu} + \boldsymbol{Q}^t(\mathbf{\Theta}) = \boldsymbol{\mu}(t+1) + \mathbf{\Theta}(\boldsymbol{x}^t - \boldsymbol{\mu}t) \tag{13}$$

with forecast error given by

$$\boldsymbol{x}^{(t+1)} - \boldsymbol{F}^{(t+1)}(\mathbf{\Theta}) = \boldsymbol{S}^{(t+1)} - \mathbf{\Theta}\boldsymbol{S}^t. \tag{14}$$

The drift terms are estimated by the sample means of the first differenced series, i.e., $\hat{\boldsymbol{\mu}} = n^{-1}\sum_{t=2}^{n}\Delta\boldsymbol{x}^t$. Therefore, $\hat{\boldsymbol{S}}^t = \boldsymbol{x}^t - \hat{\boldsymbol{\mu}}t$ and $\hat{\boldsymbol{u}}^t = \Delta\boldsymbol{x}^t - \hat{\boldsymbol{\mu}}$. The $\mathbf{\Theta}$ matrix is estimated either via reduced rank regression or via multivariate least squares.

3.2. Machine Learning Approaches

Recurrent Neural Networks (RNNs) are the most commonly used machine learning models for sequence prediction problems. Unlike standard feedforward neural networks, RNNs have feedback connections — which is biologically more realistic.

Instead of neurons, RNNs have memory blocks, also denoted as cells, that may be connected into multiple layers. A block has components that make it smarter than a classical neuron and a memory for recent sequences. The idea (illustrated in Figure 1) is that, in each layer, the same RNN block repeats for every time step ($t = 1, .., T$), sharing the same weights and biases between each of them. The feedback loop of the block helps the network to propagate the hidden state to the future time steps. The input to the block at time step t is a vector $\boldsymbol{x}^t \in \mathbb{R}^m$, being m the number of features. The output of a block is a vector $\boldsymbol{y}^t \in \mathbb{R}^n$. It is worth nothing that n is a an externally tuned hyperparameter that may take on any appropriate value. To use RNNs for time series forecasting [12], it is necessary to project the output of the block to the expected forecasting horizon k by means of a dense layer that must be connected on top of the last recurrent block.

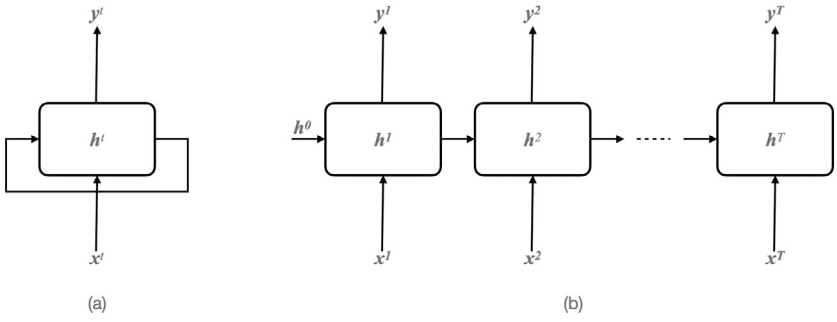

Fig. 1. Single-layer RNN: folded (a) and unfolded (b). The input, output and hidden state at time step t are denoted as $\boldsymbol{x}^t$, $\boldsymbol{y}^t$ and $\boldsymbol{h}^t$, respectively.

The most popular RNN blocks are the Elman RNN (ERNN) block [13], Long Short-Term Memory (LSTM) block [14] and the Gated Recurrent Unit (GRU) [15].

3.2.1. ERNN

At each time step t, an ERNN block is characterized by an hidden state $\boldsymbol{h}^t \in \mathcal{R}^n$ that results from the application of an activation function (the sigmod, mostly) to the input vector $\boldsymbol{x}^t \in \mathbb{R}^m$ and to the hidden state of the previous time step $\boldsymbol{h}^{(t-1)}$. Moreover, the ERNN block produces an output vector $\boldsymbol{y}^t \in \mathbb{R}^n$ that results from the application of another activation function (the hyperbolic tangent, usually) to the hidden state $\boldsymbol{h}^t$. More precisely:

$$\boldsymbol{h}^t = \sigma(\boldsymbol{W}_i \boldsymbol{h}^{(t-1)} + \boldsymbol{V}_i \boldsymbol{x}^t + \boldsymbol{b}_i) \tag{15}$$

$$\boldsymbol{y}^t = \tanh(\boldsymbol{W}_o \boldsymbol{h}^t + \boldsymbol{b}_o) \tag{16}$$

where $\boldsymbol{W}_i \in \mathbb{R}^{n \times n}$, $\boldsymbol{W}_o \in \mathbb{R}^{n \times n}$, $\boldsymbol{V}_i \in \mathbb{R}^{m \times m}$ are weight matrices, and $\boldsymbol{b}_i$, $\boldsymbol{b}_o$ are bias vectors.

The structure of the basic ERNN block is as shown in Figure 2.

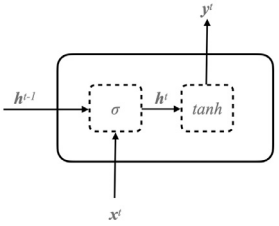

Fig. 2. Scheme of the Elman RNN block.

ERNNs have very complex dynamics and they are difficult to train. Going back with the gradients, the values may get either smaller exponentially (*vanishing gradient* problem) or larger exponentially (*exploding gradient* problem).

3.2.2. LSTM

An LSTM block features an input activation function, three gates (input, forget, output), an internal recurrence loop (the Constant Error Carousel), an output activation function and peephole connections (Fig. 3). The LSTM block has an input $\boldsymbol{x}$ (with size m) and produces an output $\boldsymbol{y}$ (with size n). The output of the LSTM block is recurrently connected back to the block input.

The LSTM block is characterized by the following weights:

- Input weights: $\boldsymbol{W}_z, \boldsymbol{W}_i, \boldsymbol{W}_f, \boldsymbol{W}_o \in \mathbb{R}^{n \times m}$
- Recurrent weights: $\boldsymbol{R}_z, \boldsymbol{R}_i, \boldsymbol{R}_f, \boldsymbol{R}_o \in \mathbb{R}^{n \times n}$
- Peephole weights: $\boldsymbol{p}_i, \boldsymbol{p}_f, \boldsymbol{p}_o \in \mathbb{R}^n$
- Bias weights: $\boldsymbol{b}_z, \boldsymbol{b}_i, \boldsymbol{b}_f, \boldsymbol{b}_o \in \mathbb{R}^n$

Let $\boldsymbol{x}^t$ be the input vector at time t. Then the vector formulas for the LSTM block forward pass can be written as:

$$\boldsymbol{z}^t = g(\boldsymbol{W}_z \boldsymbol{x}^t + \boldsymbol{R}_z \boldsymbol{y}^{t-1} + \boldsymbol{b}_z) \quad \text{input activation function} \tag{17}$$

$$\boldsymbol{i}^t = \sigma(\boldsymbol{W}_i \boldsymbol{x}^t + \boldsymbol{R}_i \boldsymbol{y}^{t-1} + \boldsymbol{p}_i \odot \boldsymbol{c}^{t-1} + \boldsymbol{b}_i) \quad \text{input gate} \tag{18}$$

$$\boldsymbol{f}^t = \sigma(\boldsymbol{W}_f \boldsymbol{x}^t + \boldsymbol{R}_f \boldsymbol{y}^{t-1} + \boldsymbol{p}_f \odot \boldsymbol{c}^{t-1} + \boldsymbol{b}_f) \quad \text{forget gate} \tag{19}$$

$$\boldsymbol{c}^t = \boldsymbol{z}^t \odot \boldsymbol{i}^t + \boldsymbol{c}^{t-1} \odot \boldsymbol{f}^t \quad \text{internal recurrence loop} \tag{20}$$

$$\boldsymbol{o}^t = \sigma(\boldsymbol{W}_o \boldsymbol{x}^t + \boldsymbol{R}_o \boldsymbol{y}^{t-1} + \boldsymbol{p}_o \odot \boldsymbol{c}^t + \boldsymbol{b}_o) \quad \text{output gate} \tag{21}$$

$$\boldsymbol{y}^t = h(\boldsymbol{c}^t) \odot \boldsymbol{o}^t \quad \text{output activation function} \tag{22}$$

where $\odot$ denotes the point-wise multiplication of two vectors, $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic sigmoid, $g(x)$ and $h(x)$ are usually the hyperbolic tangent $\tanh(x)$.

The way the LSTM block reduces the vanishing gradient problem is by creating an internal memory state which is simply added to the processed input. In this way, the multiplicative effect of small gradients is greatly reduced. The forget gate determines which states are remembered or forgotten.

Several variants of the LSTM architecture for RNNs have been proposed since its inception in 1995. A thorough survey and performance evaluation of LSTM variants was presented by Greff et al. [16], considering three representative tasks: speech recognition, handwriting recognition, and polyphonic music modeling.

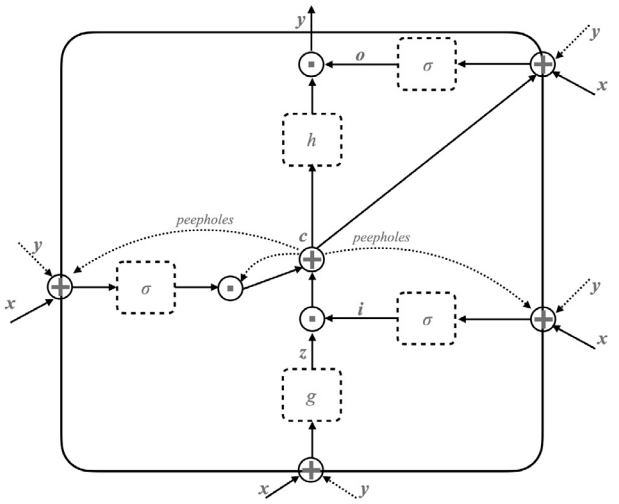

Fig. 3. The most general scheme of an LSTM block. Continuous arrows refer to vectors at time t, dashed arrows refer to vectors at time $t-1$.

In a recent work by Makridis *et al.* [17], an LSTM models is built to perform one step ahead prediction, using multiple data streams as inputs, for predictive maintenance in the context of maritime industry.

3.2.3. GRU

The GRU block, proposed by Cho et al. [15], is a simplified variant of the LSTM block. Neither peephole connections nor output activation functions are used. The input and the forget gate are coupled into an update gate. Finally, the output gate (called reset gate) only gates the recurrent connections to the block input ($\boldsymbol{W}_z$).

In a recent work by Ardeshiri and Ma [18], a GRU-based deep learning approach is used to predict the remaining useful life (RUL) of lithium-ion batteries (LIBs), accurately.

4. Experimental Evaluation

To evaluate the forecasting methods described in Section 3, we have used common Python modules like StatsModels, Tensorflow, Keras and Pandas. Regarding the Theta method, we have implemented (using R) the multivariate version, starting from the bivariate one that is provided with the book by Assimakopoulos and Nikolopoulos [11]. Our code and data are available on a public GitHub repository.[2]

The LSTM model has a first layer with $n = 100$ units and activation function *relu*. Before providing the output to the second layer, its fixed-length is repeated once for each required time step. The second layer has the same structure of the first one. Finally, a Time distributed layer of Dense type is added. The selected optimizer is *Adam*, the learning rate has been set to 0.001, the number of epochs is 100 for the FRED dataset, 70 for Air Quality, Appliances Energy Prediction and Gas Turbine, and 20 for Beijing PM2.5 Data. Using different epochs for each dataset is necessary to overcome convergence issues. The GRU and ERNN models have the same structure and hyper-parameters of the LSTM one.

We have adopted two different execution environments. The first one is the Google Cloud Platform environment owned by Sidel. The other one is the High Performance Computing facility of the University of Parma. With these platforms, the largest machine learning model we had to train (LSTM with 100 units, for the Appliances Energy Prediction dataset described below) required about 15 minutes.

[2] https://github.com/hpc-unipr/forecasting

Table 1. Dataset features

Dataset	Variables	Variables considered	Observations	Observations considered
FRED	2	2	753	753
Air Quality	13	12	9358	7410
Appliances Energy Prediction	28	27	19735	19735
Beijing PM2.5 Data	8	5	43824	41971
Gas Turbine CO and NOx Emission	11	11	7384	7384

4.1. Datasets

In Table 1 we show the main characteristics of the selected datasets. For all of them, we performed some preprocessing in order to remove categorical variables, as well as variables with a huge amount of missing data (more than half of observations). Furthermore, we removed consecutive time steps characterized by missing values of the same variables. For non-consecutive missing data we performed interpolation to reconstruct the observation values, using the na.interp R-function of the forecast library.

For all datasets, before applying the forecasting algorithms, we performed feature selection, that is, given an initial set of m features, we found the subset within $n < m$ features that is "maximally informative" about the original data.

The Federal Reserve Economic Data (FRED) dataset [19] collects data aggregated on weekly basis from 21 January 1999 to 21 June 2013 of DEXUSEU versus DEXUSUK foreign exchange rates.

The Air Quality dataset [20] contains instances of hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. The device was located on the field in a significantly polluted area, at road level, within an Italian city. Data were recorded from March 2004 to February 2005 (one year), representing the longest freely available recordings of on field deployed air quality chemical sensor devices responses.

The Appliances Energy Prediction dataset [21] collects temperature and humidity measurements in a house, at 10 min for about 4.5 months, merged together with weather information from the nearest airport weather station (Chievres Airport, Belgium). For consistency with the other datasets, we aggregated data on a hourly basis.

The Beijing PM2.5 dataset [22] contains the PM2.5 hourly data of the US Embassy in Beijing, collected between January 1st, 2010 and December 31st, 2014.

Finally, the Gas Turbine CO and NOx Emission dataset [23] collects instances of 11 sensor measures aggregated over one hour (by means of average or sum) from a gas turbine located in Turkey's north western region for the purpose of studying flue gas emissions, namely CO and NOx (NO + NO2).

In our experiments, the forecast horizon has been set based on data frequency: 13 for weekly series and 48 for hourly ones.

4.2. Experimental Results

In Table 2 we report the accuracy measures (sMAPE, MASE and OWA) of the statistical methods and the machine learning models applied to the datasets described in Table 1.

For the FRED series the ERNN model is the one that performs best for all the metrics, while the GRU performs worst. The VARMA method gives the highest accuracy results for Air Quality series in both cases (complete and reduced number of variables); for the complete dataset (12 variables), GRU presents the worst performance for all of the accuracy measures, while for the reduced dataset (11 variables) ERNN and LSTM perform worst in terms of sMAPE and MASE, respectively. VARMA method achieves the best results even as for Appliances Energy Prediction for complete and reduced dataset considering sMAPE and OWA metrics, while in terms of MASE the lowest values belong to LSTM for the complete dataset and to GRU for the reduced one. For the Beijing PM2.5 series VARMA gives the best performance in terms of sMAPE, but considering MASE the best result is achieved by ERNN model and in terms of overall metric (OWA) by the Theta method. For Gas Turbine Emission series the Naïve method is the best performing for all the metrics on complete and reduced datasets, while the worst results are reached by VARMA in terms of MASE on the complete dataset and by ERNN considering sMAPE on the reduced dataset.

Table 2. Performance of the statistical methods and machine learning models for accuracy measures (sMAPE, MASE and OWA) on the simulated time series.

FRED

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	1.193649	1.185661	1.277451	1.017414	2.541121	**1.003728**
MASE	1.380297	1.371132	1.475193	1.178984	2.911474	**1.162238**
OWA	1	0.993334	1.069478	0.8532542	2.119089	**0.8414553**

Air Quality (12 variables)

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	25.920726	**21.268016**	31.04238	108.438836	111.103119	86.857429
MASE	1.066287	**0.945339**	1.240972	1.482605	1.690928	1.019372
OWA	1	**0.8535366**	1.180707	2.786958	2.936038	2.153444

Air Quality (11 variables)

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	26.441247	**21.418943**	32.80628	79.379805	88.20006	102.380801
MASE	1.08536	**0.805769**	1.303255	1.165231	1.283546	1.610113
OWA	1	**0.7762279**	1.220741	2.037855	2.25915	2.677747

Appliances Energy Prediction (27 variables)

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	12.459746	**11.009363**	13.27577	66.989181	62.444374	64.885003
MASE	1.459115	1.246086	1.56609	**1.206099**	1.119784	1.2623
OWA	1	**0.8687979**	1.069404	3.101522	2.889565	3.036342

Appliances Energy Prediction (25 variables)

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	12.482669	**10.523774**	13.11852	82.92169	61.644413	73.088591
MASE	1.454296	1.166184	1.539047	1.472314	**1.068622**	1.264733
OWA	1	**0.8224799**	1.054608	3.827668	2.836602	3.362429

Beijin PM2.5 Data

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	19.939999	**16.872395**	19.7011	91.777397	82.140564	74.744674
MASE	2.150922	1.812714	2.123974	1.907544	1.884399	**1.515209**
OWA	1	0.8444598	**0.9877453**	2.744764	2.497738	2.226463

Gas Turbine CO and NOx Emission (11 variables)

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	**2.284616**	3.145704	2.733125	83.956744	94.877858	97.109777
MASE	**1.339281**	1.832027	1.601448	1.5264774	1.826159	1.609431
OWA	**1**	1.372413	1.196035	18.94425	21.44628	21.85383

Gas Turbine CO and NOx Emission (9 variables)

Accuracy	Naïve	VARMA	Theta	LSTM	GRU	ERNN
sMAPE	**1.843312**	2.383139	2.300286	92.796812	72.158405	102.637407
MASE	**1.298374**	1.321114	1.619854	1.6535555	1.510719	1.705797
OWA	**1**	1.155186	1.247749	25.808	20.1548	28.49738

Notes: We note with 'bold' the best performance.

5. Conclusion

In this study we have provided a comparison between some advanced statistical and machine learning methods for multi-step multivariate time series forecasting. We have evaluated the forecasting accuracy of the methods on series with different lengths, dimensions and data frequency.

An aspect we want to highlight is the outperforming of VARMA with respect to the other methods in the majority of the time series considered; anyway, it is necessary to extend the analysis to a wider range of datasets before stating its major accuracy with respect to RNN models. Among the statistical methods, the Theta method has been he worst one,

with respect to the considered dataset. However, it has always outperformed the machine learning models. Finally, our results do not allow to decide the best RNN mode among ERNN, LSTM and GRU.

We believe that this work can be a starting point for further investigation on the forecasting power of statistical and machine learning methods, with respect to multivariate multi-step time series forecasting, considering the high relevance they have in the field of predictive maintenance.

Acknowledgements

This research benefited from the HPC (High Performance Computing) facility of the University of Parma, Italy.

References

[1] Thyago P. Carvalho, Fabrízzio A. A. M. N. Soares, Roberto Vita, Roberto da P. Francisco, João P. Basto, and Symone G. S. Alcalá. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, 137:106024, 2019.

[2] Xiao-Sheng Si, Wenbin Wang, Chang-Hua Hu, and Dong-Hua Zhou. Remaining useful life estimation – a review on the statistical data driven approaches. *European Journal of Operational Research*, 213(1):1–14, 2011.

[3] Djoko Darwanto, Deny Hamdani, Didik Dwi Hariyanto, and Otto Hari Karyawan. Statistical analysis of partial discharge characteristics for predictive maintenance of generator of geothermal power plant. In *IEEE International Conference on Condition Monitoring and Diagnosis*, pages 1003–1006, 2012.

[4] Juan José Montero Jimenez, Sébastien Schwartz, Rob Vingerhoeds, Bernard Grabot, and Michel Salaün. Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics. *Journal of Manufacturing Systems*, 56:539–557, 2020.

[5] Jiaming Yin, Weixiong Rao, Kai Zhao, Mingxuan Yuan, Jia Zeng, Chenxi Zhang, JiangFeng Li, and Qinpei Zhao. Experimental study of multivariate time series forecasting models. In *28th ACM International Conference on Information and Knowledge Management*, 2019.

[6] Dimitrios D. Thomakos and Konstantinos Nikolopoulos. Forecasting Multivariate Time Series with the Theta Method: Multivariate Theta Method. *Journal of Forecasting*, 34(3):220–229, April 2015.

[7] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 Competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36:54–74, March 2020.

[8] Fotios Petropoulos et al. Forecasting: theory and practice. *arXiv:2012.03854*, 2020.

[9] John C. Chambers, Satinder K. Mullik, and Donald D. Smith. How to Choose the Right Forecasting Technique. *Harvard Business Review*, July 1971.

[10] Helmut Lütkepohl. *New Introduction to Multiple Time Series Analysis*. Springer, 2005.

[11] V. Assimakopoulos and K. Nikolopoulos. The Theta model: a decomposition approach to forecasting. *International Journal of Forecasting*, 16(4):521–530, 2000.

[12] Bryan Lim and Stefan Zohren. Time Series Forecasting With Deep Learning: A Survey. *Philosophical Transactions of the Royal Society A*, 379: 20200209, February 2021.

[13] Jeffrey Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[15] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[16] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber. LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, 2017.

[17] Georgios Makridis, Dimosthenis Kyriazis, and Stathis Plitsos. Predictive maintenance leveraging machine learning for time-series forecasting in the maritime industry. In *IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8, 2020.

[18] Reza Rouhi Ardeshiri and Chengbin Ma. Multivariate gated recurrent unit for battery remaining useful life prediction: A deep learning approach. *International Journal of Energy Research*, 45(11):16633–16648, 2021.

[19] Federal Reserve Bank of St. Louis. Federal Reserve Economic Data (FRED). https://fred.stlouisfed.org/graph/?id=DEXUSEU,DEXUSUK.

[20] S. De Vito, E. Massera, M. Piga, L. Martinotto, and G. Di Francia. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical*, 129(2):750–757, 2008.

[21] Luis M. Candanedo, Véronique Feldheim, and Dominique Deramaix. Data driven prediction models of energy use of appliances in a low-energy house. *Energy and Buildings*, 140:81–97, 2017.

[22] Xuan Liang, Tao Zou, Bin Guo, Shuo Li, Haozhe Zhang, Shuyi Zhang, Hui Huang, and Song Xi Chen. Assessing Beijing's PM2.5 pollution: severity, weather impact, APEC and winter heating. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2182):20150257, 2015.

[23] Heysem Kaya, Pinar Tüfekci, and Erdinç Uzun. Predicting co and nox emissions from gas turbines: novel data and a benchmark pems. *Turkish Journal of Electrical Engineering and Computer Science*, 27:4783 – 4796, 2019.