Revista de Sistemas de Informação da FSMA
n. 24 (2019) pp. 19-30

SISTEMAS DE INFORMAÇÃO
http://www.fsma.edu.br/si/sistemas.html

TRILHA PRINCIPAL

Prognostics Of An Industrial Gas Turbine: A Time Series Forecasting Data Driven Approach

Heron Felipe Rosas dos Santos[1], Leila Weitzel[1], Ana Paula Barbosa Sobral[1]
[1]Instituto de Ciência e Tecnologia (ICT), Universidade Federal Fluminense (UFF)

***Abstract*—Prognostics assesses and predicts future machine health, which includes detecting incipient failures and predicting remaining useful life. Several studies approached prognostics as a time series forecasting problem. The main goal of this study is to evaluate the performance of a set of methods in the prediction of future values from a dataset of time series collected from sensors installed on an industrial gas turbine. Methods tested include the use of ensembles of feedforward neural networks, ensembles of long short-term memory networks, exponential smoothing, and Auto Regressive Integrated Moving Average (ARIMA) models. Results show that the use of ARIMA models to forecast on the dataset is the best default method to apply, and is the only method that consistently beats a simple naïve no-change model.**

***Keywords*—prognostics, time series, forecasting, neural networks, ARIMA**

I. Introduction

Condition Based Maintenance (CBM) is a maintenance policy that aims to take maintenance action before a failure happens. CBM times the maintenance action by assessing product condition, and predicting failure based on data gathered from the product. Even though the technologies and technical methods for CBM are still in their infancy, advancements in information technology have accelerated growth in CBM technology by providing network bandwidth, data collection and retrieval, data analysis, and decision support capabilities for large datasets of time series. Nowadays, process data, collected in the form of time series, is often compressed and archived for record keeping and only retrieved for emergency analysis after a fault has occurred. However, this data could be of tremendous advantage when combined with effective analytics and superior computing power capable of generating knowledge from the data [1], [2].

Diagnostics and prognostics are two parts of CBM. Diagnostics is a reactive process. It takes place after a fault has already occurred and aims to determine the root cause of the failure. It cannot prevent machine downtime and the corresponding expenses. On the other hand, prognostics is a proactive process. It assesses and predicts future machine health, which includes detecting incipient failures and predicting remaining useful life [3]. When done properly, it can improve machine conditions and decrease the amount of down time for all equipments.

There are three classes to the current approaches to prognostics: model based, data driven and hybrid [4].

- Model based approaches presume that it is possible to build a mathematical model from the understanding of the physical mechanisms involved in the failure modes of the machine that bases the model. These approaches have the advantage of providing the ability to incorporate physical understanding of the system. However, if the understanding of the system degradation is poor, it may be difficult to model the system behavior.
- Data driven approaches use data gathered from sensors or by the machine operators to track features that indicate the degradation of the system. Data driven approaches can leverage computer intelligence techniques like neural networks and decision trees or statistical techniques like auto-regressive models.
- Hybrid approaches combine model based and data driven approaches in an effort to leverage the advantages and diminish the disadvantages of both types of approaches.

Historically, empirical evaluations have shown in several domains that statistically sophisticated or complex methods do not necessarily achieve more accurate forecasts for time series than simpler methods. However, recent evaluations have concluded that complex methods based on computational intelligence and neural networks have caught up, and that simple methods can no longer claim to outperform computational intelligence methods without a proper empirical evaluation [5].

Several studies on prognostics have dealt with it from a time series forecasting perspective. These studies apply diverse forecasting techniques, such as Auto Regressive Moving Average models, Recurrent and Feedforward Neural Networks, Support Vector Regression, Markov models and simple linear regression, to time series of different types of condition related parameters (e.g. machine vibration) in order to predict remaining useful life and next failure occurrence [6], [7], [8], [9], [10].

There are published works showing positive results from neural networks applied to time series forecasting in domains other than CBM. [11], [12]. These studies show that the application of time series forecasting methods based on neural networks, combined with huge amounts of historical data, may lead to better prognostics of industrial machines. Ultimately, better prognostics may lead to reduced maintenance costs and increased production availability.

Neural networks are subject to variance due to weight initialization and to the training algorithm (stochastic gradient descent). This means that even if using the same architecture and training set, different instances of the training procedure might yield different values for the network weights and, consequently, different output values for the same input. A way to deal with this variance is to employ an ensemble of networks.

The basic idea behind an ensemble is to build a prediction model by combining a collection of simpler base models [13]. Silveira and Mauá [14] applied ensembles of Information Retrieval and Word Embedding algorithms to answer multiple-choice questions. The ensemble showed better performance than previous approaches, both on accuracy and on standard deviation. Ensembles of classifiers also showed superior performance in distinguishing between Brazilian and European Portuguese, when compared with any of the individual classifiers in the ensembles [15].

The main goal of this study is to evaluate the performance of a set of methods in the prediction of future values of monitored parameters in industrial machines. The methods are data driven and deal with prognostics from a time series forecasting perspective.

This study is an empirical evaluation, executed by forecasting on a dataset collected from an industrial gas turbine. The dataset consists of a collection of time series. The study evaluates forecasts generated by neural networks with different architectures selected for each of the individual time series in the dataset. The study also evaluates forecasts generated by well-established forecasting techniques (ARIMA models and exponential smoothing). The naïve no-change model serves as a benchmark for all the considered forecasting methods.

The contribution of this paper is in the generation of knowledge directed specifically to the improvement of prognostics, when considered as a time series forecasting problem. To the best of the authors' knowledge, there is a scarce amount of published evaluations of multiple forecasting methods on data from real machines. This knowledge is useful for the understanding of the best forecasting methods available for those who want to estimate the remaining useful life of industrial machines.

The structure of the remainder of this study is as follows. Section II presents some related works in the areas of prognostics and predictive maintenance. Section III describes the dataset used in the study, the preprocessing applied to the dataset prior to any model building, and the neural networks used to generate forecasts. Section IV describes forecasting methods well established within the time series forecasting community. Section V covers the method and metric for forecast evaluation. Finally, Section VI presents the results and provides a conclusion for the study.

II. Related Work

A. Prognostics as Time Series Forecasting

Several studies approached prognostics as a time series forecasting problem. Pham et al. [6] used an Auto Regressive Moving Average (ARMA) model on baseline data. The authors used deviations from the ARMA model on future values as a degradation index. After the degradation index reaches a threshold, Pham et al. used Cox's PHM (Proportional Hazards Model) to create a survival probability curve as a function of time followed by Support Vector Regression (SVR) to predict remaining useful life.

Heng et al. [7] used an artificial neural network with the most recent values of a condition index (bandpass vibration) as inputs. The ANN does not predict future values for the condition index, rather predicting probability of failure in fixed time intervals ahead of the last condition index measure. Heng et al. benchmarked the proposed model against two ANN models predicting reliability and an Elman Recurrent Neural Network (RNN) that approached prognosis as time series prediction. The authors considered RNNs as the most commonly used models in the prognostics literature, but reported superior results from their proposed model.

Datong et al. [8] presented an on-line (incremental learning) SVR based strategy (MSPO-SVR) for prediction of industrial sensor data. The authors tested the strategy on the Tennessee Eastman benchmark dataset. Datong et al. compare the presented strategy against traditional on-line SVR. MSPO-SVR showed superior performance based on MSE.

Niu and Yang [9] used a neural network to fuse a set of features into a single value used for condition monitoring. After the condition index reaches a threshold value two non-linear techniques, Dempster-Shafer regression and least-squares support vector machines, predict the future behavior of the monitored index. A weighted average combines the predictions from both methods.

Cho et al. [10] developed a hybrid approach to predict the next failure time of a centrifugal compressor using vibration data. Bellow a threshold value, the authors applied a Markov model to predict next failure time. Above the threshold value, they apply a mix of moving average filter and simple linear regression.

The studies cited show the variety of forecasting techniques used in the field of prognostics. Many of the techniques from these studies are included in the current research. Some of these studies have limitations in the number of techniques evaluated, while some of them perform tests on benchmarks datasets, allowing comparisons with other studies. Table I provides an overview of the cited related work.

B. Time Series Forecasting With Neural Networks

There are published works showing positive results from neural networks applied to time series forecasting in domains other than CBM. Khashei and Bijari [11] introduced an approach based on using an Auto Regressive Integrated Moving Average (ARIMA) model to extract features from a time series. The features serve as training input to a single hidden layer feedforward network. Ma et al. [12] used a Long Short-Term Memory (LSTM) neural network to predict traffic speed.

C. Big Data and Predictive Maintenance

Yan et al. [16] discuss how industry 4.0 has aroused a new round of interest in advanced manufacturing and how advances in sensor technology, computer science and big data lead to a new industrial revolution. While discussing the challenges of integrating data from various sources, not only sensor data but also environmental data and staff knowledge, the authors present a case study of machining tool degradation which, like the current research, uses only sensor data, though also considering unstructured sensor data (images from a laser scanner).

TABLE I. RELATED WORK ON PROGNOSTICS

Study	Relevant Points	Forecasting Method	Data
Pham et al. [6]	• Degradation index derived from sensor data • Reliability mixed with traditional time series forecasting • No comparison to other methods	Three Stage: ARMA → PHM → SVR	Vibration data from methane compressor (same as [9])
Heng et al. [7]	• Comparison between prognostics as reliability and as time series forecasting	Feedforward Neural Network, Elmann RNN	Irving Pulp and Paper
Datong et al. [8]	• Raw sensor data • Proposed methods compared only to a different SVR method • MSE as metric	MSPO-SVR	Tennessee Eastman
Niu and Yang [9]	• Degradation index derived from sensor data using NN • No comparison to other methods	DSR, LS-SVM	Vibration data from methane compressor (same as [6])
Cho et al. [10]	• Raw sensor data • No comparison to other methods • MAPE as metric (non normalized data)	Two Stage: Markok Chain → Moving Average Filter + Linear Regression	Vibration data from gas compressor

Wan et al. [17] state that one of the challenges faced by preventive maintenance in the context of industry 4.0 is the design of the algorithm for offline prediction and evaluation of service life, and that recent technology allows for real time alarms while still lacking in predictability. The authors propose a cloud architecture that uses neural networks to predict machining tool remaining life (in fixed intervals).

III. DATA AND METHODS

The following subsections describe the dataset used in this research, the preprocessing applied to the data prior to any forecasting, and the forecasting methods based on neural networks applied to the dataset.

A. Data

The data used in this study comes from an oil platform's data management system. The system stores data from multiple sensors installed through the offshore facility. The focus of the current research is on data collected from the sensors in one of the platform's gas turbines, which is responsible for generating power for the platform.

The dataset includes values collected from 32 sensors, which measure pressure (P), vibration (V), temperature (T) and rotation speed (R). Fig. 1 shows a schematic of the instrumentation's position on the gas turbine.

The raw data also includes values logged when the turbine is not operating. These values deviate considerably from the values observed when the machine is operating. Hence, the study excludes those data points which were collected during stoppages.

The data management system does not log values for each sensor at a fixed sampling rate. Different reasons may trigger a

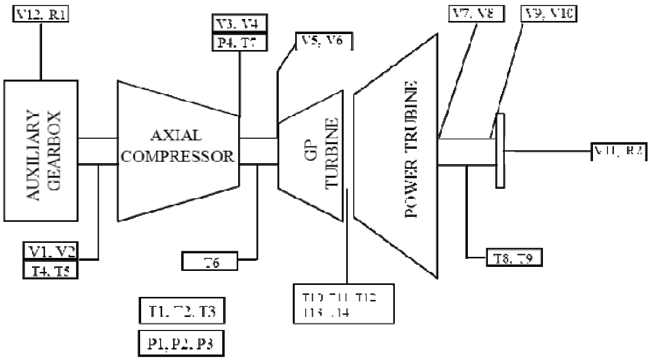


Fig. 1 - Position of embedded sensors on the turbine under consideration.

data logging event for each sensor, resulting in asynchronous time series. In order to allow the research to proceed with the use of standard forecasting techniques for evenly spaced time series, preprocessing of the dataset aggregates the time series into evenly spaced data.

Daily bins divide the dataset. For each time series, the aggregated values equal the mean of any values collected during a 24 hours period (a daily bin). In the event there are not any values collected in a 24 hours period for one of the series, making it impossible to calculate an average value for that day, this day with no data uses a value linearly interpolated from the adjacent days. Working with daily values is a research design choice. Daily values are useful for monitoring long term degradation mechanics of a machine. The research could easily be redone with different intervals, leading to different results.

In order to guarantee secrecy of the real operating parameters, normalization of the dataset using a standard scaler occurs. The resulting series have mean zero, standard deviation one and are dimensionless. Regardless of real value hiding, input normalization is a common preprocessing step for neural networks. The resulting dataset contains 1461 daily values for each of the 32 sensors.

We perform a preliminary analysis of the study's dataset. The first step in the analysis is the augmented Dickey-Fuller test with the objective of assessing trend stationarity. The second step is Levene's test with the objective of assessing homoscedasticity (stationarity on variance). The final test used in the preliminary analysis is the Anderson-Darling test for normality. These tests allow for the analysis of some relevant characteristics of the time series. The test results are shown in section IV.

The study splits the dataset in two. The first 90% of the data serves as the training dataset for all of the forecasting methods. The final 10% of the data is the test set, used in order to evaluate the accuracy of the models in proper out-of-training-sample data. There are no changes to model parameters in the evaluation phase. All forecasting methods output one-step ahead forecasts.

The choice of the 10% size for the test set is due to the nature of the current research's data. With cross sectional data (data collected at a single point in time, like images of dogs or a collection of forms filled by a group of people), the data have no obvious ordering and the test set selection is random in the available data. That makes the test set and the training set similarly distributed, even for bigger sizes of test sets. For time series, given that the data is collected sequentially over time, there is an obvious ordering to the data, and the test set comes from the most recent observations. If the test set is too big, there can be significant differences in patterns between the training set and the test set.

It would not be feasible to follow a more sophisticated version of training/test sets split, like time series cross validation [18], due to the increase required in the number of instances of the neural network training problem (146 times increase for the one-step ahead forecast scenario). Hence, the authors use a 10% size for the test set as a tradeoff between computational resources and better forecasts.

B. Feedforward Neural Networks

Neural network is a term that encompass a large class of models and learning methods. Neural networks are nonlinear statistical models that model the outputs as nonlinear functions of linear combinations of the inputs. One builds a neural network by connecting simple computing cells called neurons or processing units. This study uses neural networks implemented in Python using the Keras library [19].

There are three basic elements to a neuron's model. First, a set of connecting links to other neurons, each characterized by a weight of its own. Second, an adder, often called a propagation function, used to sum all the input signals to the neuron. The third element is the activation function. The activation function limits the output of the neuron and is responsible for the nonlinearities in the network. Equations (1) and (2) give the output x_k of a neuron k that uses weighted sum as its adder. Haykin [20] gives further details on the mathematics of neural networks.

$$v_k = \sum_{j=1}^{m} w_{kj}x_j + b_k \#(1)$$

Where:
w_{kj}: Weight of the connection between neuron j and neuron k;
x_j: Output of neuron j;
b_k: The bias of neuron k.

$$x_k = \varphi(v_k)\#(2)$$

Where:
v_k: Activation potential of neuron k;
φ: Is the activation function.

One type of neural network used is this study is Feedforward Neural Networks (FNN), in which there are no loops. The layers are ordered and a network layer only uses as input the output from the previous layer. This study creates an independent FNN for each time series in the dataset, which uses as input only lagged values from the time series it forecasts. This means this study does not considers possible information in the interaction between series.

It is necessary to define the architectures of the FNNs before proceeding with the final training. All FNNs used in this study use the same procedure for architecture selection. The architecture selection procedure starts with a split of the training set: 2/3 for training and 1/3 for validation.

At this point, it is beneficial to make a clear distinction between the validation set and the test set. The validation set is a subdivision of the training set. The use of a validation set is an approach to the selection of hyperparameters of a machine-learning algorithm. Hyperparameters that control model capacity (like number of units in the hidden layer) should not be learned on the training set. Otherwise, the hyperparameters would always choose maximum model capacity, resulting in overfitting. This is the importance of using a validation set for hyperparameter selection [21]. In the case of a neural network, these hyperparameters are the network parameters that the backpropagation algorithm cannot learn: number of layers, number of units in each layer, form of activation function, between others.

This study considers several architectures with different numbers of units in the hidden layer and different numbers of lagged values as inputs. This study only considers architectures with a single hidden layer. The procedure continues with training of the different architectures on the reduced training set for a maximum of 1000 epochs with early stopping if the validation loss (Mean Squared Error (MSE), as in (3)) does not improve after 10 consecutive epochs. The final FNNs, trained on the entire training set, use the architectures that presented the smallest average validation loss after 10 training runs. All FNNs use hyperbolic tangent as the activation function of the units in the hidden layer.

$$MSE = \frac{\sum_{t=1}^{k}(\hat{y}_t - y_t)^2}{k} \#(3)$$

Where:
$\hat{y}_t$: Forecasted value for y at time t;
y_t: Actual value for y at time t;
k: Number of forecasted points.

After selection of the architectures for each time series, the forecasts created for the test set use a collection of 8 FNNs that have the same previously selected architecture. Averaging combines the outputs from the individual FNNs. The authors expect this procedure will reduce variance in the results. The size 8 for the network collections is a design choice by the authors. According to the central limit theorem, variance would be further reduced by increasing the number of networks in the collection [22]. However, since this research does not deal with confidence intervals, we elect to use 8 networks in an effort to achieve a representative result, while keeping computational costs low.

C. *Long Short Term Memory Units*

A Recurrent Neural Network (RNN) is a neural network that allows feedback loops. The state of an RNN, which consists of the activation values of the hidden units, depends on the past values of the state. The presence of feedback loops helps RNNs in processing sequential data, like time series. RNNs are susceptible to the problem of gradient instability. The longer the network runs, the more unstable are the gradients on inputs further back in time.

Long Short Term Memory (LSTM) units, are a special type of processing unit used to build the hidden layers in a LSTM neural network. LSTM units address the problem of gradient instability by creating paths through time that have derivatives that will neither vanish nor explode [21]. The LSTM units have an adaptive forget gate designed to reset a unit's state when its contents are no longer relevant. The forget gate controls the weight of the state self-loop and, in that way, how much of the information in the state is preserved or discarded between time steps. Gers et al. [23] give further detail on LSTM units.

All LSTM networks tested use only a single time lagged value as input, given that unrolling the computational graph shows that the output of a recurrent network is actually a function of the entire time series and that the LSTM units should have the capability to accumulate all relevant past information in their states. This study creates an independent LSTM network for each series in the dataset. The only input of each independent LTSM is the value at $t-1$ of the time series it forecasts.

It is necessary to define the number of LSTM units in the hidden layer before proceeding with the final training. All LSTM networks used in this study use the procedure that follows for selection of the number of units in the hidden layer, which is similar to the procedure used for the feedforward architecture described in the previous section.

The procedure starts with a split of the training set: 2/3 for training and 1/3 for validation. This study considers several architectures with different number of units in the hidden layers. The procedure continues with training of the networks on the reduced training set for a maximum of 1000 epochs with early stopping if the validation loss (MSE) does not improve after 10 consecutive epochs. The reason to use a validation set for LSTM network size selection is the same as stated on past section on FNN networks. The final LSTM networks, trained on the entire training set, uses the number of hidden units that presented the smallest average validation loss after 10 training runs.

After selection of the architectures for each time series, the forecasts created for the test set use a collection of 8 LSTM networks that have the same previously selected architecture. Averaging combines the outputs from the individual LSTM networks. The authors expect this procedure will reduce variance in the results. The same comment from last section on the size of the network collections applies.

IV. Well Established Forecasting Methods

The following subsections describe well-established methods developed by practicing forecasters. This section also describes the naïve no-change forecast, which is the simplest of all forecasting methods. The naïve forecast is the minimum a forecasting method must be able to improve upon.

A. *Exponential Smoothing*

Exponential smoothing is a forecasting approach that uses all historical values as predictors, giving more weight to more recent values, as in (4) for Simple Exponential Smoothing (SES). The equation shows that the forecast for time $t+1$ is a weighted average between the most recent observation x_t and the forecast for time t. Recursively substituting $\hat{x}_t$ yields (5).

$$\hat{x}_{t+1} = \alpha x_t + (1-\alpha)(\hat{x}_t)\#(4)$$

Where:
$\hat{x}_{t+1}$: Forecasted value for x at time $t+1$;
x_t: Actual value for x at time t;
α: Smoothing parameter.

$$\hat{x}_{t+1} = \alpha x_t + \alpha(1-\alpha)x_{t-1} + \alpha(1-\alpha)^2 x_{t-2} + \cdots \#(5)$$

As long as $0 < \alpha < 1$, the weight given to each observation decreases exponentially as each observation comes from further in the past, hence the name exponential smoothing.

SES is a suitable forecasting method for data without trend or seasonal patterns. SES has a flat forecast function, meaning the forecast for all subsequent time steps take the same value [18]. There are several exponential smoothing methods other than SES. It is possible for the ES method to allow for trend and/or seasonal components. Hyndman et al. [24] identify a total of fifteen exponential smoothing methods, with five possibilities for the trend component (None, Additive, Additive damped, Multiplicative and Multiplicative damped) and three possibilities for the seasonal component (None, Additive and Multiplicative).

This study follows Hyndman et al. [24] state space approach for ETS (Error, Trend, Seasonality) model selection. An ETS model is a statistical model that underlies an ES method. Each individual time series on the dataset has an ETS model selected for itself. The R function *ets()* applies Hyndman's approach automatically. It is the computational tool used for ES method selection and parameter estimation.

B. *Auto Regressive Integrated Moving Average Models*

Auto Regressive Integrated Moving Average (ARIMA) models combine Auto Regressive (AR) models, Moving Average (MA) models and differencing. Differencing is a way to make time series stationary by computing the differences between consecutive observations. The addition of differencing allows for non-stationary (on trend) data. An ARIMA (p,d,q) model combines an AR model of order p, a MA model of order q and d order differencing.

The study uses the R function auto.arima() for model selection and parameter estimation. It conducts a search over possible models and selects the best one based on the lowest Akaike Information Criterion (AIC). Hyndman and Khandakar [25] provide details on the function's implementation.

C. *Naïve Forecast*

A naïve model is a model that presumes things will remain the same as they have in the past. For time series data, the naïve (no change) model simply forecasts the next observation to be equal as the latest observation. The naïve model serves as a benchmark model for other models. If a model cannot produce better forecasts than a simple alternative like naïve no-change, it is of no use [26].

V. Forecasting and Forecast Evaluation

Forecast accuracy assessment occurs after model training. The metric used for this model evaluation phase is the same used in model training: MSE, as in (3). The accuracy assessment uses the test set consisting of the last 10% of the full dataset.

The model evaluation phase tests model accuracy not only on one-step ahead forecasts, but also on multistep ahead forecasts. Forecasting windows tested are 1, 2, 5, 7, 10 and 14 days ahead.

For ARIMA models and ETS (with additive errors), parameter estimation uses maximum likelihood, which results in similar parameter estimates to minimizing the sum of squared errors for these models. The error is the difference between the fitted (forecasted) value and the actual value at a given time t. The forecasting equations for ARIMA and ETS models for a given time $T + h$, where T is the last observed value in the time series, depend on the forecasted values for time steps between T and $T + h$. This means the forecasting equations for ARIMA and ETS are inherently recursive and a single model can be used to predict a forecasting window of arbitrary size [18].

For models based on neural networks (LSTM NN and FNN), the forecasting strategy depends on the architecture of the network. One possibility is to map a k step ahead forecast to a network with k units in its output layer, as in the architectures discussed in [27]. Other approach is to create a network with a single unit in the output layer, and to apply this network recursively, the forecast for time $t + 1$ serves as input for the model to forecast the values at time $t + 2$ and so on, until all required forecasts are computed. This is the approach used in the Neural Network Auto Regression described in [18]. In this research, we elect to use the second approach since it is more similar to the way forecasting in ARIMA and ETS works and, more importantly, this approach means a single network can be used to forecast in windows of any size.

Retraining the neural networks after the observation of every new sample in the test set would require significant computational resources. In order to avoid the computational costs, there are no updates to network weights during the model evaluation stage. In order to test the different forecasting methods in the same conditions, there are also no changes to the parameters of the ARIMA and ETS models, even though the computational costs would be significant smaller for these methods.

The presented MSE are the average for all of the time series. The results do not show what the best method for each individual univariate time series would be. They show which method would deliver the best results, on average, for a random univariate time series drawn from the dataset, which in turn consists of a diverse collection of time series collected from the same industrial turbine.

VI. Results And Discussion

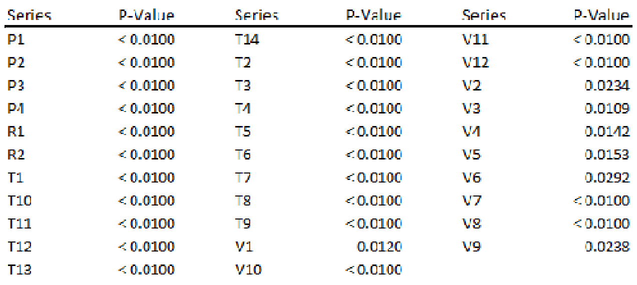
A. *Preliminary Analysis of the Time Series*

Table II bellow shows the results (p-values) for the augmented Dickey-Fuller test (alternative hypothesis of stationarity) for the 32 time series in the dataset. Table III shows the p-values for Levene's test (alternative hypothesis of heteroscedasticity) and Table IV shows the p-values for the Anderson-Darling test. As these tables show, with a 5% significance level, it is possible to say that all of the 32 time series in the dataset are trend stationary, heteroscedastic and not normally distributed.

B. *ARIMA Model Selection*

ARIMA models assume that the data is homoscedastic. However, as per the last subsection, the dataset shows heteroscedastic behavior. One option is this situation is to ignore the violation of the assumption of homoscedasticity. Another option is to apply a transformation to the data. We consider both options. For each individual time series, we fit on the training data models with and without a Box-Cox transform and select the model with the lowest AIC.

TABLE II. P-VALUES FROM THE AUGMENED DICKEY-FULLER TEST FOR EACH OF THE TIME SERIES IN THE DATASET



Series	P-Value	Series	P-Value	Series	P-Value
P1	< 0.0100	T14	< 0.0100	V11	< 0.0100
P2	< 0.0100	T2	< 0.0100	V12	< 0.0100
P3	< 0.0100	T3	< 0.0100	V2	0.0234
P4	< 0.0100	T4	< 0.0100	V3	0.0109
R1	< 0.0100	T5	< 0.0100	V4	0.0142
R2	< 0.0100	T6	< 0.0100	V5	0.0153
T1	< 0.0100	T7	< 0.0100	V6	0.0292
T10	< 0.0100	T8	< 0.0100	V7	< 0.0100
T11	< 0.0100	T9	< 0.0100	V8	< 0.0100
T12	< 0.0100	V1	0.0120	V9	0.0238
T13	< 0.0100	V10	< 0.0100		

TABLE III. P-VALUES FROM LEVENE'S TEST FOR EACH OF THE TIME SERIES IN THE DATASET

Sensor	P-Value	Sensor	P-Value	Sensor	P-Value
P1	< 0.0001	T14	< 0.0001	V11	< 0.0001
P2	< 0.0001	T2	< 0.0001	V12	< 0.0001
P3	< 0.0001	T3	< 0.0001	V2	< 0.0001
P4	< 0.0001	T4	< 0.0001	V3	< 0.0001
R1	< 0.0001	T5	< 0.0001	V4	< 0.0001
R2	< 0.0001	T6	< 0.0001	V5	< 0.0001
T1	< 0.0001	T7	< 0.0001	V6	< 0.0001
T10	< 0.0001	T8	< 0.0001	V7	< 0.0001
T11	< 0.0001	T9	< 0.0001	V8	< 0.0001
T12	< 0.0001	V1	< 0.0001	V9	< 0.0001
T13	< 0.0001	V10	< 0.0001		

TABLE IV. P-VALUES FROM THE ANDERSON-DARLING TEST FOR EACH OF THE TIME SERIES IN THE DATASET

Sensor	P-Value	Sensor	P-Value	Sensor	P-Value
P1	< 0.0001	T14	< 0.0001	V11	< 0.0001
P2	< 0.0001	T2	< 0.0001	V12	< 0.0001
P3	< 0.0001	T3	< 0.0001	V2	< 0.0001
P4	< 0.0001	T4	< 0.0001	V3	< 0.0001
R1	< 0.0001	T5	< 0.0001	V4	< 0.0001
R2	< 0.0001	T6	< 0.0001	V5	< 0.0001
T1	< 0.0001	T7	< 0.0001	V6	< 0.0001
T10	< 0.0001	T8	< 0.0001	V7	< 0.0001
T11	< 0.0001	T9	< 0.0001	V8	< 0.0001
T12	< 0.0001	V1	< 0.0001	V9	< 0.0001
T13	< 0.0001	V10	< 0.0001		

The *auto.arima()* function is capable of applying a Box-Cox transformation to the data. The selection of the λ parameter is automatic. The model fitting considers a 7 days seasonality period. Table V shows the results obtained on the test set by the ARIMA models selected with this procedure (labeled ARIMA transform). Table V also shows the results obtained on the test set by ARIMA models that do not consider seasonality and that do not allow for any previous data transform (labeled simple ARIMA).

TABLE V. AVERAGE MEAN SQUARED ERRORS OBTAINED ON THE TEST SET BY ARIMA MODELS

Method	Days in Forecast Window					
	1	2	5	7	10	14
ARIMA Transform	0.17225	0.23766	0.37852	0.42262	0.46479	0.50207
Simple ARIMA	0.16538	0.22355	0.33863	0.37346	0.40702	0.43645

The results show that the use of simple ARIMA models delivers better results than considering seasonality and a using a data transform to deal with the heteroscedasticity. Table VI shows for how many of the time series a method delivers the best results. For 19 time series both procedures select the same model. The Table shows the distribution of the remaining 13 series between both ARIMA model selection procedures.

TABLE VI. FOR HOW MANY SERIES AN ARIMA SELECTION PROCEDURE DELIVERS THE BEST RESULTS.

Method	Days in Forecast Window					
	1	2	5	7	10	14
ARIMA Transform	6	4	4	5	6	6
Simple ARIMA	7	9	9	8	7	7

As the tables show, not accounting for seasonality and prior data transformation, which is not a good practice, does delivers better results on average and for a bigger number of time series. Hence, we elect to ignore the heteroscedasticity of the data and to consider the best possible ARIMA models as the models without a data transform and that do not consider seasonality. Table VII shows the orders of the models selected for each of the time series.

C. Neural Network Architecture Selection

The prediction methods that use independent neural networks require selection of hyperparameters. As previously discussed, the hyperparameters selection phase uses a split of the training set: 2/3 for training and 1/3 for validation. Table VIII shows the architectures considered for each independent FNN. The number of time lags on an FNN equates to the number of units on the input layer. The independent LSTM networks tested use a single time lag as input and test the same number of units in the hidden layer as the considered FNNs. Both FNNs and LSTM networks have a single unit on the output layer, representing the forecast for time $t + 1$. Table IX shows the selected FNN architectures and Table X shows the selected LSTM architectures for each time series, based on the average validation loss after 10 training runs.

TABLE VII. ORDERS OF THE ARIMA MODELS SELECTED FOR EACH OF THE TIME SERIES

Sensor	Model	Sensor	Model
P1	ARIMA(0,1,3)	T6	ARIMA(1,1,1)
P2	ARIMA(1,1,3)	T7	ARIMA(1,1,1)
P3	ARIMA(5,1,4)	T8	ARIMA(1,1,1)
P4	ARIMA(1,1,1)	T9	ARIMA(3,1,1)
R1	ARIMA(2,1,1)	V1	ARIMA(1,1,2)
R2	ARIMA(1,1,1)	V10	ARIMA(1,1,2)
T1	ARIMA(1,1,2)	V11	ARIMA(0,1,2)
T10	ARIMA(3,1,3)	V12	ARIMA(2,1,2)
T11	ARIMA(2,1,1)	V2	ARIMA(1,1,2)
T12	ARIMA(2,1,1)	V3	ARIMA(1,1,0)
T13	ARIMA(2,1,1)	V4	ARIMA(2,1,0)
T14	ARIMA(2,1,1)	V5	ARIMA(2,1,1)
T2	ARIMA(1,0,1)	V6	ARIMA(3,1,0)
T3	ARIMA(1,1,2)	V7	ARIMA(1,1,2)
T4	ARIMA(2,1,1)	V8	ARIMA(1,1,2)
T5	ARIMA(0,1,4)	V9	ARIMA(1,1,3)

TABLE VIII. ARCHITECTURES CONSIDERED FOR EACH INDEPENDENT NEURAL NETWORK.

Time Lags	1, 2, 3, 4, 5
Hidden Layer Units	1, 2, 3, 4, 5, 8, 10, 15, 20

TABLE IX. SELECTED FNN ARCHITECTURES FOR EACH TIME SERIES.

Time Series	P1	P2	P3	P4	R1	R2	T1	T10	T11	T12	T13	T14	T2	T3	T4	T5
Hidden Layer Units	3	8	10	10	1	3	5	8	8	15	5	8	15	8	15	20
Time Lags	1	4	1	1	5	3	3	2	2	5	2	2	1	1	3	3

Time Series	T6	T7	T8	T9	V1	V10	V11	V12	V2	V3	V4	V5	V6	V7	V8	V9
Hidden Layer Units	10	4	8	10	15	20	15	5	20	15	20	15	15	15	15	5
Time Lags	2	1	2	3	2	2	3	3	2	1	2	3	1	3	1	2

TABLE X. SELECTED LSTM ARCHITECTURES FOR EACH TIME SERIES.

Time Series	P1	P2	P3	P4	R1	R2	T1	T10	T11	T12	T13	T14	T2	T3	T4	T5
Hidden Layer Units	20	5	3	8	20	20	8	8	5	4	8	10	20	4	20	15

Time Series	T6	T7	T8	T9	V1	V10	V11	V12	V2	V3	V4	V5	V6	V7	V8	V9
Hidden Layer Units	20	4	15	15	15	20	20	1	2	1	1	1	1	2	5	1

D. Network Averaging Effect on Variance

In order to evaluate the variance on the final models, and the effect of averaging on reducing this variance, 8 sets of 8 networks each were created, both for the FNN and for the LSTM networks, all with the same architecture, as selected in the previous subsection. The boxplots in Fig. 3 thru Fig. 6 are for the test set errors, for the 1 and the 14 days forecasting windows, both for a single network (64 in total, composing the 8 sets) and for the 8 sets of 8 networks. From the boxplots, it is visible that averaging reduces significantly the variance in the results. It should be noted that there are outliers in the boxplots for the ensembles (both LSTM NN and FNN) in the 1 day ahead window. This indicates that the size 8 for the ensembles, while clearly reducing the variance on the final results, is not enough to ensure that the results fall well within the middle of the distribution in small forecasting window.

Table XI for the standard deviation of the test errors, both for the single networks and for the sets of 8 networks, shows further evidence on the effect of averaging. As expected, averaging does reduce the variance on the final result, while having no negative impact on the expected result, as seen by the limited effect on the median test error (Table XII). Averaging is especially important in the case of LSTM networks, where outliers are present on the boxplots for a single network in all forecasting windows.

TABLE XI. STANDARD DEVIATION OF THE TEST ERRORS. ALL FORECASTING WINDOWS.

Method	Days in Forecast Window					
	1	2	5	7	10	14
1-FNN	0.00721	0.00520	0.00488	0.00884	0.01547	0.02157
FNN ensemble	0.00131	0.00076	0.00091	0.00239	0.00372	0.00550
1-LSTM	0.01572	0.00916	0.00911	0.00940	0.01079	0.01407
LSTM ensemble	0.00561	0.00353	0.00357	0.00369	0.00430	0.00566

TABLE XII. MEDIAN OF THE TEST ERRORS. ALL FORECASTING WINDOWS.

Method	Days in Forecast Window					
	1	2	5	7	10	14
1-FNN	0.21285	0.25897	0.35422	0.39680	0.46560	0.54325
FNN ensemble	0.21232	0.25876	0.35443	0.39770	0.46180	0.54177
1-LSTM	0.20397	0.24774	0.33601	0.36917	0.41097	0.45813
LSTM ensemble	0.20348	0.24799	0.33557	0.36926	0.41162	0.45897

E. Results on the Test Set

Table XIII summarizes the results obtained by applying the proposed forecasting methods to the test set. For both the FNN and LSTM networks, the presented results are from a single collection of 8 networks with the same architecture. Averaging combines the outputs from the 8 networks. The selection of the collection between the 8 sets previously discussed is random.

TABLE XIII. MSE ON THE TEST SET FOR EACH OF THE FORECASTING METHODS.

Days in Forecast Window	Test Dataset Loss - Mean Squared Error				
	FNN	LSTM	ARIMA	ES	Naive
1	0.21205	0.20268	0.16538	0.18055	0.17894
2	0.25821	0.24684	0.22355	0.25041	0.25484
5	0.35408	0.33462	0.33863	0.40570	0.43541
7	0.39678	0.36805	0.37346	0.45626	0.49433
10	0.46147	0.40925	0.40702	0.50454	0.54906
14	0.54359	0.45441	0.43645	0.54369	0.59119

The collection of LSTM networks yields better results than the FNNs in all forecasting windows. This result is consistent with the literature that points that recurrent neural networks are the best neural networks to process sequential data. The collection of FNNs beats the naïve forecast in forecasting windows bigger than two days. For forecasting windows bigger than one day, the univariate LSTM is capable of delivering better results than the no-change naïve method.

Exponential smoothing applied to each time series only produces worse forecasts than the naïve method in the one-step ahead forecast scenario. However, for bigger forecasting windows, the results delivered by the method quickly deteriorate, and ES loses to at least one of the collections of neural networks. We must mention that the exponential smoothing method selected for all of the time series is Simple Exponential Smoothing, meaning that the ES forecasts are constant lines.

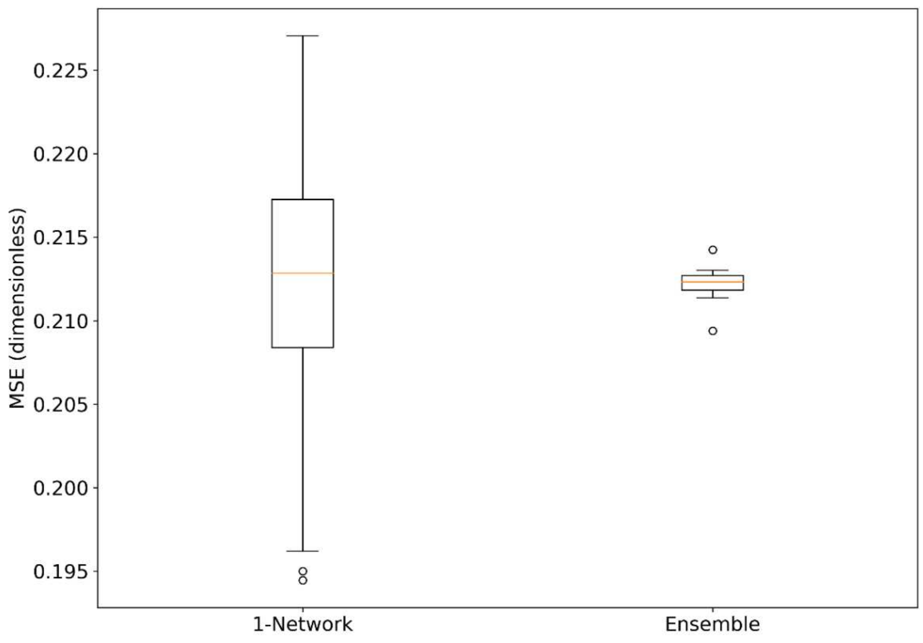


Fig. 3 - Boxplots of test errors (MSE) for FNNs in the 1 day forecasting window. Both for the test errors of a single network (1-Network) and for the average errors of 8 networks (Ensemble). Dots represent outliers.

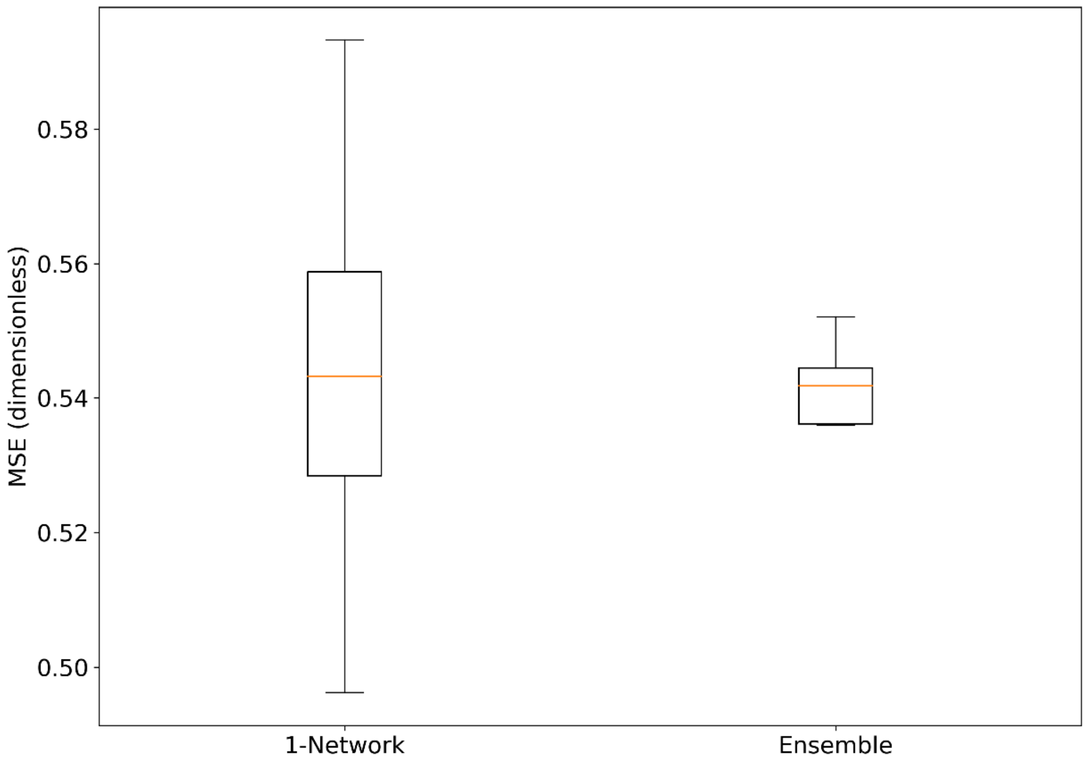


Fig. 4 - Boxplots of test errors (MSE) for FNNs in the 14 days forecasting window. Both for the test errors of a single network (1-Network) and for the average errors of 8 networks (Ensemble). Dots represent outliers.

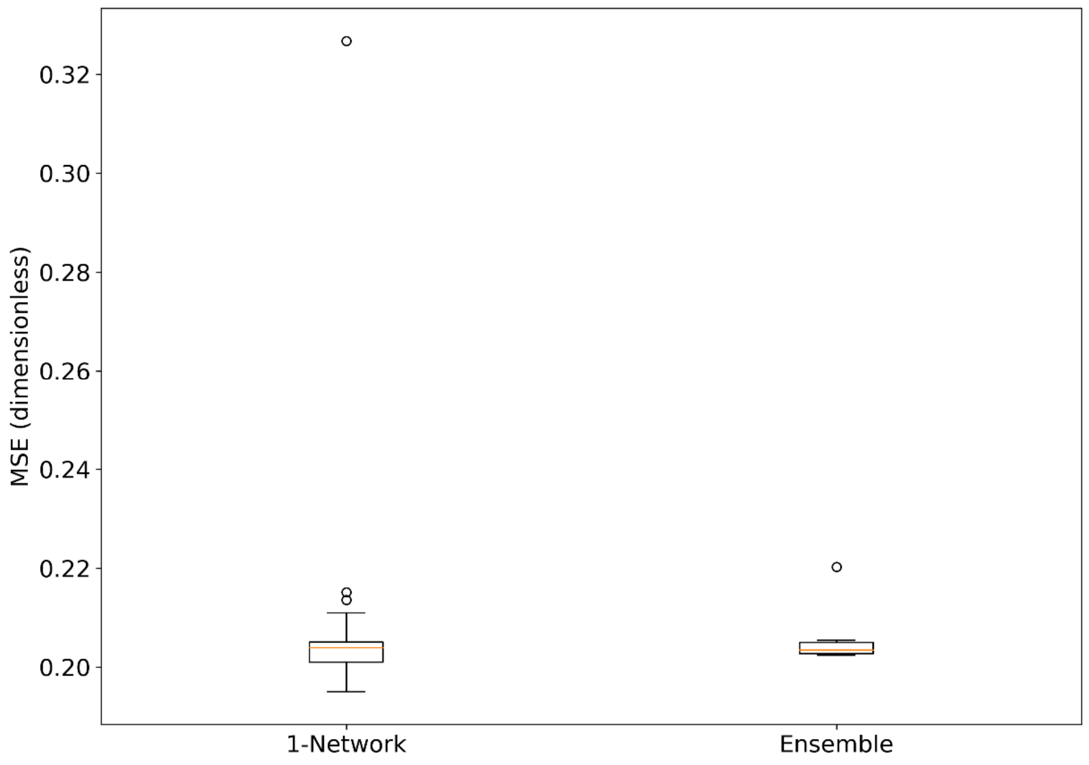


Fig. 5 - Boxplots of test errors (MSE) for LSTMs in the 1 day forecasting window. Both for the test errors of a single network (1-Network) and for the average errors of 8 networks (Ensemble). Dots represent outliers.

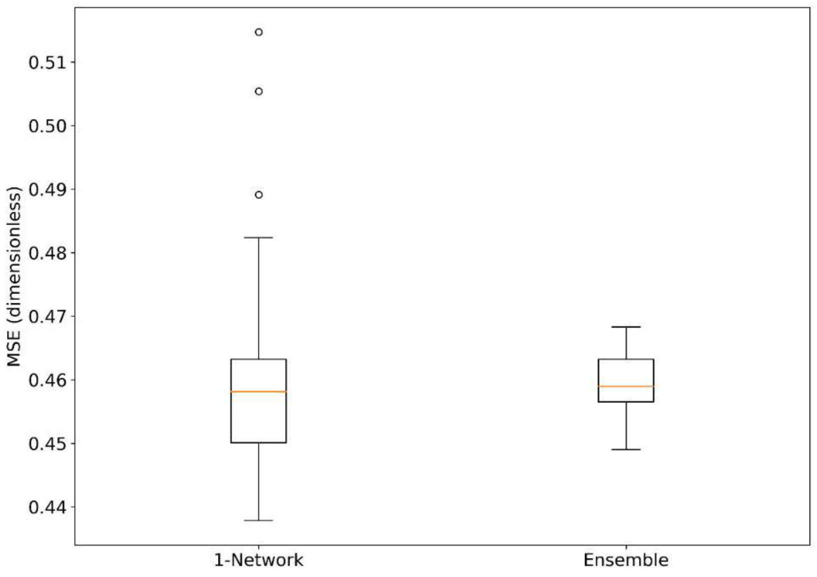


Fig. 6 - Boxplots of test errors (MSE) for LSTMs in the 14 days forecasting window. Both for the test errors of a single network (1-Network) and for the average errors of 8 networks (Ensemble). Dots represent outliers.

The method that presented the best results in this evaluation is the use of ARIMA models selected for each time series using Hyndman’s approach for model order selection. The use of ARIMA models consistently beats the naïve forecasts, and only produces slightly worse results than the LSTM networks in two forecasting windows (5 and 7 days). The best ARIMA models ignore seasonality and prior data transformation.

The fact that, for some scenarios, the naïve forecasts provide the best results might be surprising to some readers. With a 5% significance, we rejected the aDF test’s null hypothesis of unit root for all of the 32 time series in the dataset, which means the series in the dataset do not look like random walks. However, after looking at the results for the forecasts in the test set, we tried KPSS tests in the dataset. The

KPSS test has a null hypothesis of stationarity. With a 5% significance level, we only do not reject the null hypothesis of stationarity for one of the time series in the dataset (as per Table XIV). We have opposing evidences on whether the time series are random walks or not. Naïve forecasts are the optimal forecasts for random walks and the KPSS tests might explain why the naïve forecasts yield the best results in some scenarios.

TABLE XIV. P-VALUES FROM THE KPSS TEST FOR EACH OF THE TIME SERIES IN THE DATASET

Sensor	p-Value	Sensor	p-Value	Sensor	p-Value
P1	0.0100	T14	0.0100	V11	0.0100
P2	0.0100	T2	0.1000	V12	0.0100
P3	0.0100	T3	0.0100	V2	0.0100
P4	0.0133	T4	0.0100	V3	0.0100
R1	0.0100	T5	0.0100	V4	0.0100
R2	0.0100	T6	0.0100	V5	0.0100
T1	0.0100	T7	0.0100	V6	0.0100
T10	0.0100	T8	0.0100	V7	0.0100
T11	0.0100	T9	0.0100	V8	0.0100
T12	0.0100	V1	0.0100	V9	0.0100
T13	0.0100	V10	0.0100		

Besides the KPSS tests, further evidence in the results that it is difficult to model any regularity in the studied time series are that all series are forecasted with SES, which uses a flat forecast function (like the naïve model), and that the use of a transform does not benefit ARIMA forecasts. The results reinforce the difficulty in forecasting irregular time series.

Based on the dataset used on this study, should one select a single method between the ones tested in this study to produce forecasts on a diverse collection of time series drawn from industrial machinery sensors, the default forecasting method should be the use of independent ARIMA models selected for each time series. Not only ARIMA models produced the best forecasts in this empirical evaluation, but also the computational resources required to select the model parameters are much smaller.

F. Conclusion

This work empirically evaluated the forecasting performance of a set of different forecasting methods in a dataset of time series drawn from industrial sensors. The dataset comes from sensors installed in a gas turbine located on an oil platform.

We draw a few conclusions from the results. First, the results achieved show that using ARIMA models to forecast the time series is the best default methodology to apply, and is the only methodology that consistently beats a simple naïve no-change model. Second, the results add further evidence to the literature that recurrent architectures are superior to feedforward architectures for neural networks in dealing with time series data.

Third, this study also shows the positive effect on variance of averaging the output from several neural networks. The variance reduction is particularly important for individual LSTM networks, which show outliers in test set errors in all the tested forecasting windows. We elected to use collections of 8 networks on this work. This number proved not big enough to eliminate outliers in the boxplot of test set errors for the collections in small forecasting windows. Lastly, the results showed that it can be difficult to beat a naïve forecast with irregular time series that may be considered as random walks. Some of the time series in the dataset (like the rotations R1 and R2) are far less relevant to machine protection than others (like radial vibrations V1 thru V10).

This study presents limitations. First, there was a focus on evaluating neural networks. This limited the time available to evaluate other forecasting methods developed by the computer science (e.g., Support Vector Regression) and statistics communities. Moreover, there is no guarantee that it would not be possible to find neural networks capable of yielding better forecasts than the ones achieved by the best performing methods in this research. Second, while the dataset consists of 32 time series, they were drawn from only 4 types of machine sensors. Also, the time series in the dataset proved to be irregular. Some of the time series in the dataset (like the rotations R1 and R2) are far less relevant to machine protection than others (like radial vibrations V1 thru V10). Perhaps, controlling the more relevant series using the less relevant and control variables could lead to more well behaved and forecastable time series.

This research does not consider outlier removal and interactions between variables. Finally, it should be noted that on this study we focused on finding a general best forecasting method capable of dealing with all the different time series. A procedure capable of selecting a different, and presumably best, forecasting method for each of the time series could possibly yield better results on the test set.

Future studies should focus on improving the variety of the time series in the dataset, assessing a greater variety of forecasting methods, drawing better performance of models based on neural networks, considering the interactions between the different series and assessing the effect of outlier removal on model accuracy. Other research points of interest are defining a methodology to size the necessary number of networks in a collection in order to assure that result randomness is properly reduced, and a procedure capable of selecting the best forecasting method for each individual time series.

REFERENCES

[1] S. J. Qin, “Process data analytics in the era of big data,” *AIChE Journal*, vol. 60, no. 9, pp. 3092–3100, 2014.

[2] J.-H. Shin and H.-B. Jun, “On condition based maintenance policy,” *Journal of Computational Design and Engineering*, vol. 2, no. 2, pp. 119–127, 2015.

[3] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, “Prognostics and health management design for rotary machinery systemsreviews, methodology and applications,” *Mechanical systems and signal processing*, vol. 42, no. 1-2, pp. 314–334, 2014.

[4] O. E. Dragomir, R. Gouriveau, F. Dragomir, E. Minca, and N. Zerhouni,“Review of prognostic problem in condition-based maintenance,” in *Control Conference (ECC), 2009 European*. IEEE, 2009, pp. 1587– 1592.

[5] S. F. Crone, M. Hibon, and K. Nikolopoulos, “Advances in forecasting with neural networks? empirical evidence from the nn3 competition on time series prediction,” *International Journal of forecasting*, vol. 27, no. 3, pp. 635–660, 2011.

[6] H. T. Pham, B.-S. Yang, T. T. Nguyen et al., “Machine performance degradation assessment and remaining useful life prediction using proportional hazard model and support vector machine,” *Mechanical Systems and Signal Processing*, vol. 32, pp. 320–330, 2012.

[7] A. Heng, A. C. Tan, J. Mathew, N. Montgomery, D. Banjevic, and A. K. Jardine, “Intelligent condition-based prediction of machinery

reliability," *Mechanical Systems and Signal Processing*, vol. 23, no. 5, pp. 1600–1614, 2009.

[8] L. Datong, P. Yu, and P. Xiyuan, "Online adaptive status prediction strategy for data-driven fault prognostics of complex systems," in *Prognostics and System Health Management Conference (PHM-Shenzhen)*, 2011. IEEE, 2011, pp. 1–6.

[9] G. Niu and B.-S. Yang, "Intelligent condition monitoring and prognostics system based on data-fusion strategy," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8831–8840, 2010.

[10] S. Cho, J.-H. Shin, H.-B. Jun, H.-J. Hwang, C. Ha, and J. Hwang, "A study on estimating the next failure time of compressor equipment in an offshore plant," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[11] M. Khashei and M. Bijari, "An artificial neural network (p, d, q) model for timeseries forecasting," *Expert Systems with applications*, vol. 37, no. 1, pp. 479–489, 2010.

[12] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[13] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1.

[14] I. C. Silveira and D. D. Mauá, "Advances in automatically solving the enem," in *7th Brazilian Conference on Intelligent Systems (BRACIS).* IEEE, 2018, pp. 43–48.

[15] D. Castro, E. Souza, and A. L. de Oliveira, "Discriminating between brazilian and european portuguese national varieties on twitter texts," in *5th Brazilian Conference on Intelligent Systems (BRACIS).* IEEE, 2016, pp. 265–270.

[16] J. Yan, Y. Meng, L. Lu and L. Li, "Industrial Big Data in an Industry 4.0 Environment: Challenges, Schemes, and Applications for Predictive Maintenance," in *IEEE Access*, vol. 5, pp. 23484-23491, 2017.

[17] J. Wan et al., "A Manufacturing Big Data Solution for Active Preventive Maintenance," in *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2039-2047, Aug. 2017.

[18] R. J. Hyndman, and G. Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[19] F. Chollet et al., "Keras," https://keras.io, 2015.

[20] S. Haykin, *Neural networks and learning machines*. Pearson Upper Saddle River, NJ, USA:, 2009, vol. 3.

[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[22] D. C. Montgomery and G. C. Runger, *Applied statistics and probability for engineers*. John Wiley & Sons, 2010.

[23] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[24] R. Hyndman, A. B. Koehler, J. K. Ord, and R. D. Snyder, *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.

[25] R. J. Hyndman, Y. Khandakar et al., *Automatic time series for forecasting: the forecast package for R. Monash University*, Department of Econometrics and Business Statistics, 2007, no. 6/07.

[26] J. S. Armstrong, *Principles of forecasting: a handbook for researchers and practitioners*. Springer Science & Business Media, 2001, vol. 30.

[27] Z. Tang and P. A. Fishwick, "Feedforward neural nets as models for time series forecasting," *ORSA journal on computing*, vol. 5, no. 4, pp. 374–385, 1993.