

Original Research Article

Performance Analysis of Bagging Feed-Forward Neural Network for Forecasting Building Energy Demand

ABSTRACT

Forecast models play a fundamental role in anticipating the energy demand effects of buildings, households, and communities. A forecast model for anticipating from one to three days every 30 min of the building energy demand is presented. In this model, a feed-forward artificial neural network (ANN) is combined with bootstrap aggregation techniques, using a Box–Cox transformation, seasonal and trend decomposition using loess, and a moving block bootstrap technique. An analysis was conducted using the data provided by a building's energy demand; the data were collected during a period of four months, with readings every 10 s and averages of the values obtained every 30 min. The feed-forward neural-network method combined with bootstrap aggregation techniques consistently outperformed the forecasting accuracy of the original feed-forward neural network through cross-validation in the root mean square error (RMSE) and the mean absolute percentage error. From cross-validation in-sample period, used for the initial parameter estimation and model selection, it is concluded that a feed-forward neural network with the original data gives a slightly lower RMSE compared with the data generated by bootstrapped versions. However, in the cross-validation out-of-sample period used to evaluate forecasting performance, it has better consistency throughout all horizons for the ANN model combined with bootstrap aggregation techniques than for the ANN original model. The results are statistically significant according to the Ljung–Box test, which verifies that the forecast errors are not correlated and validates the proposed model.

Keywords: Feed-Forward, Neural Networks, Energy Demand Forecasting, ANN, Bootstrap, MBB

1. INTRODUCTION

Environmental concerns, such as global warming and climate change, together with other factors, such as the growing demand for electricity and the adoption of the precept of sustainable development, encourage the process of reducing the use of fossil fuels [1]. The growth in electricity demand is especially relevant in the residential sector, which currently consumes approximately 40% of global energy resources and generates about one-third of greenhouse gas emissions [2]. In this sector, there is also a marked variability in demand resulting from variations in daily and seasonal environmental conditions, which generates problems of simultaneity between demand and energy production [3].

Therefore, modeling and forecasting the energy consumption of buildings will play a fundamental role if urban areas are to reduce their overall energy consumption. Accurate modeling and forecasting of building energy demand enable numerous energy management and efficiency applications, such as demand response (DR) programs [4], urban energy infrastructure planning, estimating improvements to building energy performance, informing early-stage design decisions, and optimizing building heating, ventilation, and air conditioning systems [5].

Conventionally, building-scale electricity demand appraisals have been made using engineering software packages based on data processing of structural, geometric, and material building properties. The difficulty in reaching and validating such information is a problem for wide-scale energy forecasting [5]. In response, there is a growing interest in statistical and machine-learning techniques that have proved to be more accurate and fast approaches and that forgo the demanding

input requirements of theoretical formulations in favor of a practical set of historically recorded time series energy data [6].

Most of the works found in the literature have in common that they seek an economic goal. However, the incentives and economic sanctions established by the network operators reflect technical issues, such as line congestion or network stability. A recent, systematic review that identified 50 different methods between 1985 and June 2017 was presented by Kuster et al. [7], who identified three main approaches in terms of noteworthiness: time series models, regression-based formulations, and artificial neural networks (ANNs).

ANN methods have been widely used to forecast electric demand, by virtue of their superior performance. These models allow complex nonlinear relationships between the response variable and its predictors [8]. The current disadvantage is that data collection is deepened, which creates a broader data record, with a greater particularity level and, consequently, a higher noise level [9]. The noise level influences the ANN learning process, decreasing the capacity for generalization and causing an excess of training (overfitting). However, this effect can be reduced by stabilizing the variance and previous smoothing of the time series with bootstrap techniques [10].

In this article, the performance analysis of a univariate autoregressive model based on ANN structure to forecast building energy demand is presented, and the effect of variance stabilization and previous smoothing of the time series is demonstrated in the results. The objective is to appraise the performance of the model to reduce forecasting errors, leading to more real and high-performance solutions for decision making.

In view of the combinatorial essence of the problem, the technical implications are evaluated using a neural network autoregression (NNAR) model, which is a feed-forward neural network with one hidden layer. As previously mentioned, it is integrated with bootstrap techniques that make it possible to generate a new time series similar to the observed series, which it will use in the training of the NNAR model.

2. METHODOLOGY

Next, the different points of the research methodology applied are described. Initially, a Box–Cox transformation is made to the historical data, followed by seasonal and trend decomposition using loess (STL). The remaining component is initialized using the moving block bootstrap (MBB). After bootstrapping the remainder, the trend and seasonality are combined with the bootstrapped remainder, and the Box–Cox transformation is inverted. For each horizon, the final resulting forecast is calculated from the mean of the forecasts from the single models of the neural network. Fig. 1 gives an illustration of a methodology sequence.

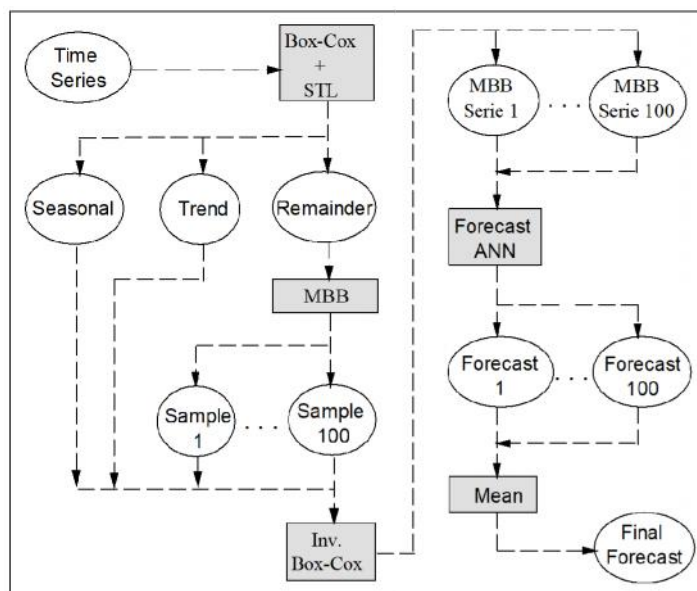


Fig. 1. Illustration of methodology sequence

2.1 Data collection

This study uses the data of the building energy demand (kW) collected in the state of Rio de Janeiro, Brazil (Fig. 2). The data were collected during a period of four months (the first four months of 2017), with readings every 10 s and averages of the values obtained every 30 min. The collected data were initially stored in the measuring device and subsequently transferred to a computer.

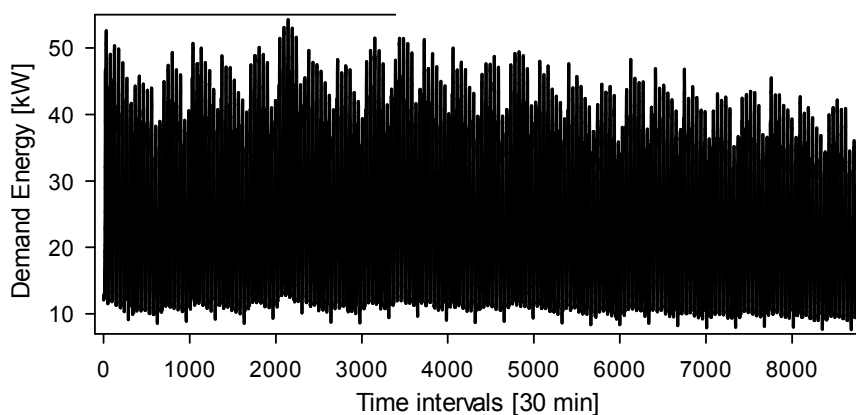


Fig. 2. Building energy demand

2.2 NEURAL-NETWORK MODELS

Artificial Intelligence is an application specialty of computer science in almost all domains of science and technology. Given the great importance of the forecast models, artificial neural networks have become an active area of research [11]. Studies in the literature show models in relation to the ANN, such as for the model of the two-phase deviation factor of a gas condensate fluid [12], forecasting of oilfield scale formation [13], and application of neural networks for forecasting the workability of self-compacting concrete [14]. Evaporation from a reservoir in semiarid environments has been estimated using the ANN and climate-based models [15]. The modeling accuracy of the aerodynamic curve of a wind turbine has been improved using neural networks [16].

A study of the selection of input variables to the ANN for forecasting hospital inpatient flows [17] has been done. An artificial network was used for forecasting water uptake under shallow saline groundwater conditions [18]. An ANN model was developed for the comparative study of different general circulation models (GCMs) (a GCM is a type of climate model) for streamflow forecast [19]. Neural network models were used for forecasting the wellhead pressure–flow-rate relationship for Niger Delta oil wells [20]. In this work, a model related to the univariate feed-forward neural network is presented, under the hypothesis that this univariate model, when combined with bootstrap aggregation techniques, can obtain a better forecast performance of building energy demand.

Recent work has been presented on forecasting the load of daytime cooling energy for institutional buildings [18] using this univariate feed-forward neural network method, in which the results were optimal with regard to performance measurements. With respect to the bootstrap approach, there exist research studies of integration with exponential models and seasonal autoregressive integrated moving average [21], [22], where the forecast performance was consistently improved.

2.2.1 Neural-network architecture

ANNs are forecasting methods that are based on simple mathematical models of the brain for information processing [8]. A neural network can be regarded as a network of neurons that are in neat layers. The inputs form the bottom layer and the outputs (forecasts) form the top layer. There may also be intermediate layers holding hidden neurons. The coefficients associated with these predictors are called “weights.” The weights are selected in the neural network setting using a learning algorithm. When adding an intermediate layer, the neural network becomes nonlinear.

That means it is a multilayer feed-forward network, in which each layer of nodes receives inputs from the preceding layers. Here, each layer of nodes receives inputs from the previous layers, while the outputs of the nodes in one layer are inputs for the next layer. The inputs for each node are combined using a weighted linear interaction, as expressed in Equation (1) [8].

$$z_j = b_j + \sum_{i=1}^4 w_{i,j} x_i \quad (1)$$

where z_j are the inputs of node j , and b_j and $w_{i,j}$ are parameters found during the training of learning from the observed data of the time series and random initial weights that are subsequently updated from the observed data [23]. The result of the linear combination is then modified by a nonlinear function before the output of each layer using a sigmoid. It is denoted in Equation (2) [8].

$$s(z) = \frac{1}{1+e^{-z}} \quad (2)$$

where $s(z)$ is the sigmoid function, and e is the base of the mathematical constant of the natural logarithm.

2.2.3 Neural-network autoregression

With time series data, lagged values of the time series can be used as inputs to a neural network. It is an NNAR model and is expressed in Equation (3).

$$y(t) = f \{y(t-1), y(t-2), \dots, y(t-p)\} \quad (3)$$

where $y(t)$ is the value of the series at time t , and p is the number of lags.

Multilayer feed-forward neural networks are denoted as NNAR (p, P, k) [m] to indicate that there are p nonseasonal lagged inputs. The optimal value is chosen according to the AIC-Akaike information criterion. Thus, for P seasonal lagged inputs, the suggested value is one, and k neurons in the hidden layer — when not specified, $k = (p + P + 1) / 2$, approaching to the nearest integer value. M represents the frequency. Because the data collection is obtained with readings every 10 s and averages of the values every 30 min, the daily frequency equals 48 [8].

In R, the NNAR algorithm is available by way of the nnetar () function [24], and it is necessary to install the forecast package [25]. Fig. (3) gives the forecast one day in advance, taking as inputs the first seven days of the data collection.

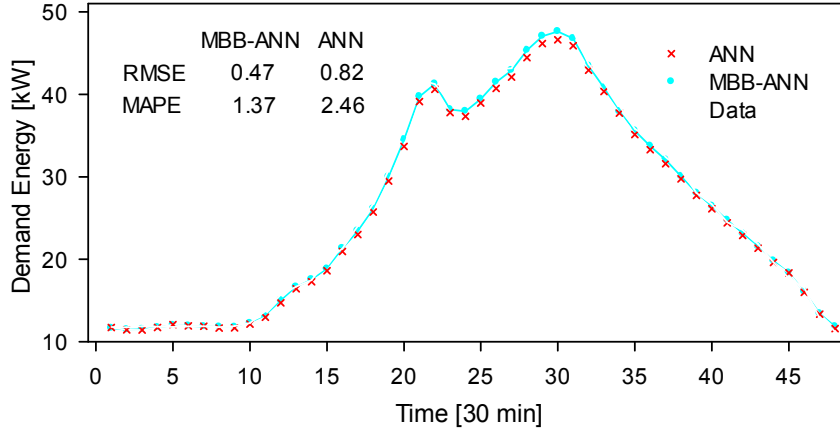


Fig. 3. Forecast one day in advance

2.3 ESTIMATING FORECASTING ERROR AND CROSS-VALIDATION

A forecast error is the difference between an observed value and its forecast; this means the unpredictable part of an observation [8]. It is denoted in Equation (10).

The two most commonly used scale-dependent measures are based on absolute errors or square errors and are given by the mean absolute percentage error (MAPE) and root mean square error (RMSE), respectively. These are denoted in Equations (4) and (5).

$$MAPE = \left(\frac{\sum_{t=T+1}^{T+h} \left| \frac{\hat{y}_t - y_t}{y_t} \right|}{h} \right) \times 100\% \quad (4)$$

$$RMSE = \sqrt{\frac{\sum_{t=T+1}^{T+h} (\hat{y}_t - y_t)^2}{h}} \quad (5)$$

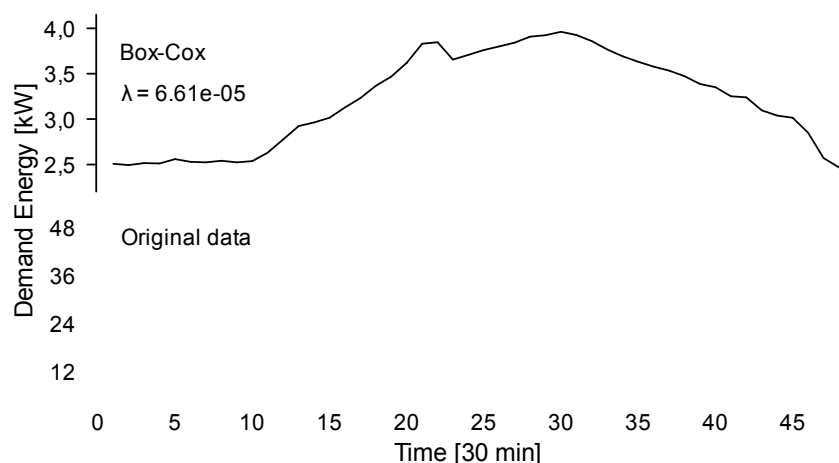
One of the most robust standard procedures performed for model evaluation in regression is K-fold cross-validation (CV). When machine-learning methods are used for the forecast, CV is appropriate to control overfitting the data. CV can appropriately control overfitting in this enforcement. If the models underfit the data and lead to strongly correlated errors, the CV method to be prevented as in equal a case they may output an underestimation of the error. However, this occurrence can be easily perceived by checking the residuals for serial correlation utilizing the Ljung–Box test [26]. The Ljung–Box test is implemented in R in the Box.test function.

2.4 THE BOX–COX TRANSFORMATION

A useful family of transformations that includes both logarithms and power transformations is the family of Box–Cox transformations, which are determined by the parameter λ [8]. These are commonly used transformations for stabilizing the variance of a time series and were originally proposed by Bergmeir et al. [27]. The transformation is defined as follows.

$$\omega_t = \begin{cases} \log(y_t), & \lambda=0; \\ (y_t^\lambda - 1)/\lambda, & \lambda \neq 0. \end{cases} \quad (6)$$

198



199

200

201

202

Fig. 4. Box–Cox transformation for the first 48 records

203

204

205

206

207

208

209

210

211

212

2.5 STL DECOMPOSITION

213

214

215

216

217

218

Time series data can show a diversity of patterns, and it is frequently fruitful to split a time series into different components, each representing a subjacent pattern category. STL is a versatile and robust method for decomposing time series [28]. In STL, loess is used to divide the time series into their trend, seasonal, and remainder components. The division is additive, summing the parts to assign the original series anew, as given in Equation (7) [8].

219

$$Y_t = S_t + T_t + R_t \quad (7)$$

220

221

222

223

224

where Y_t is the time series, S_t is the seasonal component, T_t is the trend-cycle component, and R_t is the remainder component, all at period t .

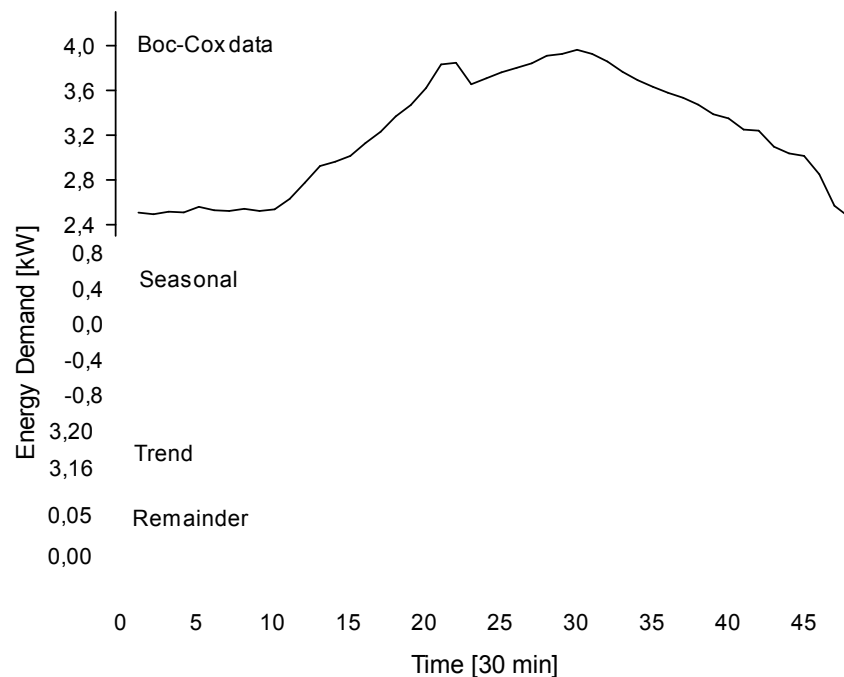


Fig. 5. STL decomposition for the first 48 records

In detail, the steps applied across STL decomposition are [22]: (i) detrending, (ii) cycle-subseries smoothing, (iii) low-pass filtering of smoothed cycle-subseries, (iv) detrending of the seasonal series, (v) deseasonalizing the original series, using the seasonal component calculated in the preceding steps, and (vi) smoothing the deseasonalized series to obtain the trend component. In R, the STL algorithm is available by way of the `stl()` function. It uses it with its default parameters, i.e., the degrees for the loess fitting are $d = 1$ in steps (iii) and (iv), and $d = 0$ in step (ii). Fig. (5) gives an STL decomposition for the time series Box-Cox transformation presented in Fig. (4).

2.6 BOOTSTRAPPING THE REMAINDER

As the time series are typically autocorrelated, a requirement is a stationarity for the forecast, which is achieved by initializing the remainder of the STL decomposition by MBB technique. For this purpose, the same procedure described in previous work [22] is followed, where, for a series of length n , with a block size of l , $[n / l] + 2$ blocks of the remaining series of an STL decomposition are indicated; then, discard a random number of values, between zero and $(l - 1)$, from the beginning of the bootstrapped series.

Finally, to obtain a series with an identical length to that of the original series, discard just as many possible values as necessary to achieve the needed length. This procedure ensures that the bootstrapped series does not necessarily start or finish on a block boundary. The number of bootstrapped versions to generate is equal to 100 [22]. Fig. (6) gives an example corresponding to the first 48 records, which includes the previous steps of Box-Cox transformation and STL decomposition.

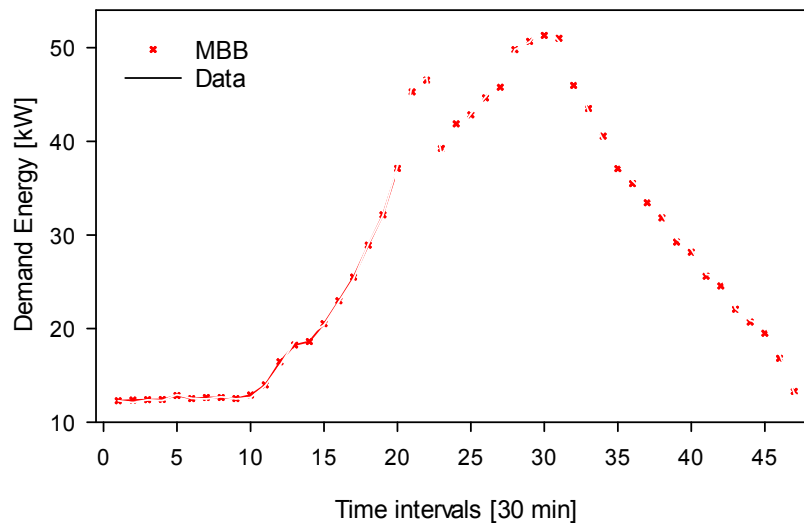


Fig. 6. Mean of bootstrapped versions for the first 48 records

3. RESULTS AND DISCUSSION

The results discussed in this section are based on the architecture of the feed-forward neural network presented in Fig. (7). The inputs in the input layer $E_i, E_{i+1}, E_{i+2}, \dots, E_{i+n}$, are consecutive data of the building energy demand, and $E_{(i+n)+1}$ is the output layer or estimate of the forecast. $H_1, H_2, H_3, \dots, H_n$ are neurons in the hidden layer.

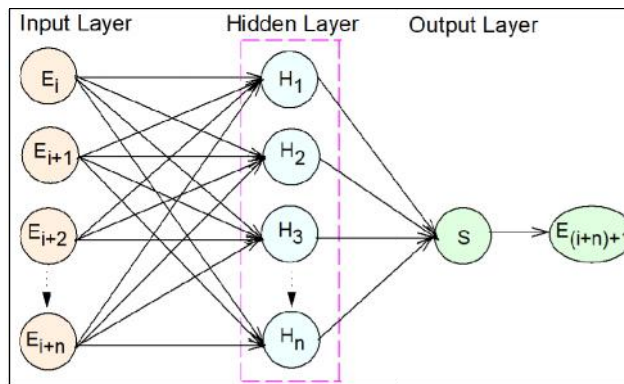


Fig. 7. Neural-network architecture

3.1 FORECASTING ONE DAY IN ADVANCE

Fig. (8) shows the result of the cross-validation one day in advance every half hour. For the first test, the data from the first seven days were used; for the second test, the data from the first eight days were used. It continued in the same way until day 119 of a total of 120 days of sample data. The next 48 values (one day) of every subseries were utilized to evaluate performance with the forecast data. For the cross-validation in-sample period used for the initial parameter estimation and model selection, the feed-forward neural network with the original data gives a slightly lower RMSE compared with the data generated by bootstrapped versions.

This difference may be because the bootstrap series generated is slightly smooth, giving rise to a slightly higher RMSE for the training of the model. However, the cross-validation out-of-sample period used to evaluate forecasting performance has better consistency throughout the horizon for the ANN model combined with bootstrap aggregation techniques than that for the ANN original model.

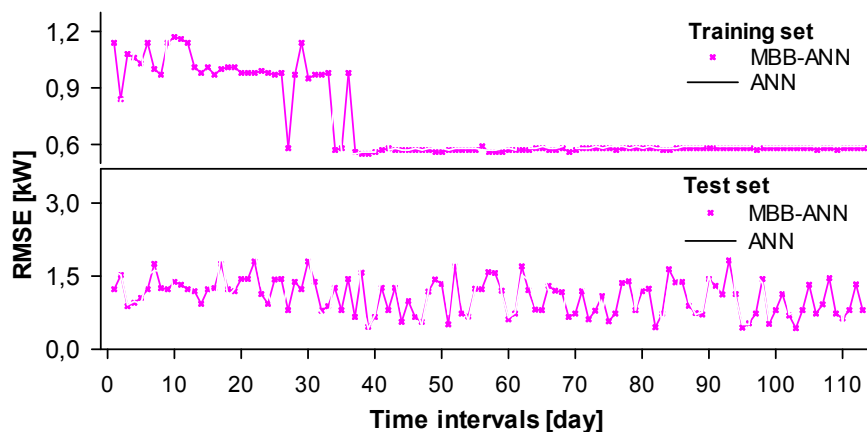


Fig. 8. Cross-validation of one-day forecast

3.2 FORECASTING TWO DAYS IN ADVANCE

For the first test, the first 15 days were utilized for training the neural network, and the first 16 days for the second test, successively, until completing the 118 days throughout the horizon of the sample data. The next 96 values (two days) of every subseries were utilized to evaluate performance with the forecast data. Fig. (9) shows that the ANN model combined with bootstrap aggregation techniques outperforms consistently the original ANN method.

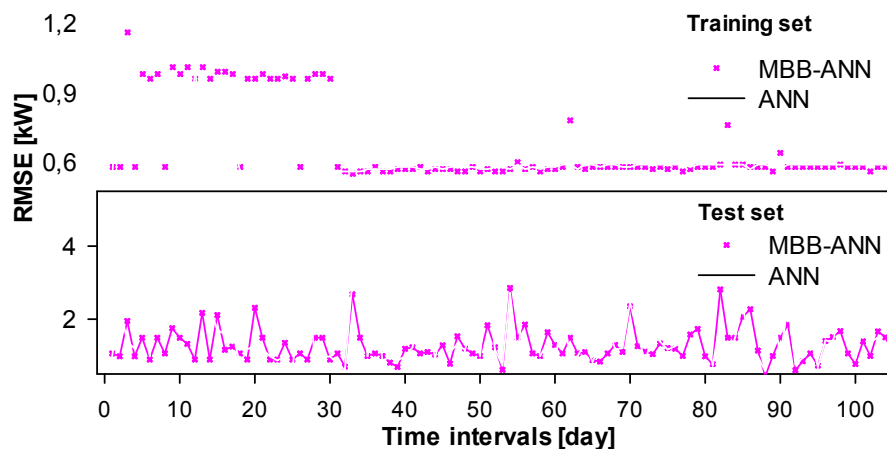


Fig. 9. Cross-validation of two-day forecast

3.3 FORECASTING THREE DAYS IN ADVANCE

The result of the cross-validation for three-day forecasts is shown in Fig. (10). For the first test, the first 21 days were utilized for training the neural network, and the first 22 days for the second test, successively, until completing the 117 days throughout the horizon of the sample data. The next 144 values (three days) of every subseries were utilized to evaluate performance with the forecast data. Of the cross-validation in-sample period for the initial parameter estimation and model selection, the feed-forward neural network with the original data gives an RMSE slightly smaller than the model of training by bootstrapped versions.

Nevertheless, as in the two previous cases, for the cross-validation out-of-sample period used to evaluate forecasting performance, the ANN model combined with bootstrap aggregation techniques has better consistency throughout the horizon than that for ANN original model, in this case, with a very significant difference.

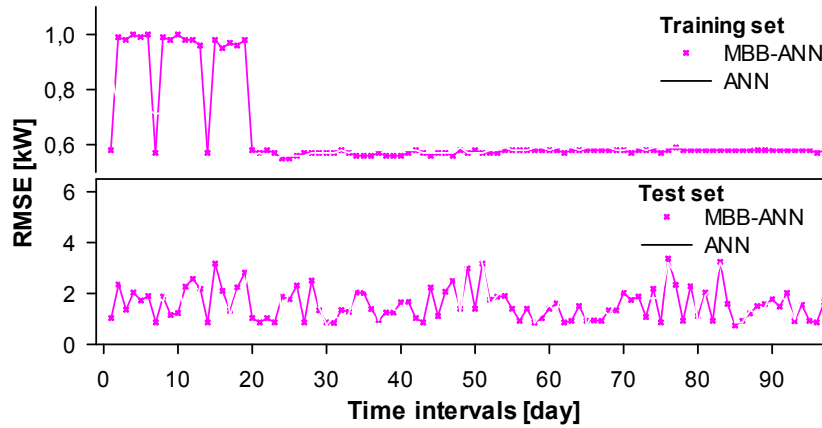


Fig. 10. Cross-validation of three-day forecast

3.4 MEAN OF K-FOLD CROSS-VALIDATION AND LJUNG–BOX TEST

Table 1 shows the mean results of the K-fold cross-validation and Ljung–Box test. For the cross-validation in-sample period used for the initial parameter estimation and model selection, the feed-forward neural network with the original data and those generated by bootstrapped versions have a similar value, slightly lower for the original data, possibly because of the smoothing obtained when using that bootstrap technique; however, the effect occurs for forecasts.

For the forecasts one day in advance, the RMSE was reduced from 1.37 to 1.09 (20 % forecast error) because of the bootstrap techniques. For forecasts two days in advance, the ANN model combined with bootstrap techniques increased slightly the forecast error compared with the previous scenario (one day); however, the ANN original model that utilized the original data had a significant forecast error. For the forecast three days in advance, the ANN method that utilized the original data can be considered inconsistent; the opposite happens with the bootstrap techniques combined with the ANN model, where there was a slight increase in forecast error with each additional day added to the forecast.

Table 1. Results of the K-fold cross-validation and Ljung–Box test

Model Set	MBB-ANN						ANN					
	Training			Test			Training			Test		
Day forecast	1	2	3	1	2	3	1	2	3	1	2	3
RMSE	0.71	0.69	0.64	1.09	1.28	1.57	0.65	0.62	0.60	1.37	1.58	2.27
MAPE	1.97	1.89	1.78	3.38	3.76	4.55	1.92	1.82	1.77	4.23	4.71	6.50
Ljung–Box test				0.98	0.98	0.97						

Considering the superior performance of the results of the ANN model combined with bootstrap techniques, it can be concluded that they do not lead to strongly correlated errors and that the RMSE obtained is not the product of systematic underestimation of the error. The mean of the Ljung–Box test of all the subseries of the K-fold cross-validation for the bootstrap techniques combined with the ANN model has a value close to the unit for the three cases studied; thus, the Ljung–Box test shows that the results are statistically significant, verifying that the forecast errors are not correlated and validating the proposed model.

4. CONCLUSION

Artificial intelligence is an application specialty of computer science in almost all domains of science and technology. In this study, the performance of the feed-forward neural-network model combined with bootstrap techniques to forecast building energy demand was evaluated using a Box–Cox transformation, STL decomposition, and MBB.

The ANN method combined with bootstrap aggregation techniques consistently outperformed forecasting accuracy compared with the original ANN method in RMSE and MAPE measurement on forecasts from one to three days of anticipation. The results are shown to be statistically significant by the Ljung–Box test, which verifies that the forecast errors are not correlated and validates the proposed model.

REFERENCES

1. M. Rabbani and M. Dolatkah, "Integration of Demand-Side Management Programs and Supply-Side Alternatives for Decentralized Energy Planning: An Analysis of Energy Import and Export Effects Integration of Demand-Side Management Programs and Supply-Side Alternatives," IGI Global, pp. 252–270, 2018.
2. M. L. Chalal, M. Benachir, M. White, and R. Shrahily, "Energy planning and forecasting approaches for supporting physical improvement strategies in the building sector: A review," *Renew. Sustain. Energy Rev.*, vol. 64, pp. 761–776, Oct. 2016.
3. F. Yutaka Kuwabata Takigawa, R. Cavalcante Fernandes, E. Antonio Cardoso Aranha Neto, D. Tenfen, and E. Taghori Sica, "Energy Management by the Consumer with Photovoltaic Generation: Brazilian Market," *IEEE Lat. Am. Trans.*, vol. 14, no. 5, pp. 2226–2232, May 2016.
4. Z. Shao, F. Chao, S.-L. Yang, and K.-L. Zhou, "A review of the decomposition methodology for extracting and identifying the fluctuation characteristics in electricity demand forecasting," *Renew. Sustain. Energy Rev.*, vol. 75, pp. 123–136, Aug. 2017.
5. R. K. Jain, K. M. Smith, P. J. Culligan, and J. E. Taylor, "Forecasting energy consumption of multi-family residential buildings using support vector regression: Investigating the impact of temporal and spatial monitoring granularity on performance accuracy," *Appl. Energy*, vol. 123, pp. 168–178, Jun. 2014.
6. C. Deb, F. Zhang, J. Yang, S. E. Lee, and K. W. Shah, "A review on time series forecasting techniques for building energy consumption," *Renew. Sustain. Energy Rev.*, vol. 74, pp. 902–924, Jul. 2017.
7. C. Kuster, Y. Rezgui, and M. Mourshed, "Electrical load forecasting models: A critical systematic review," *Sustain. Cities Soc.*, vol. 35, pp. 257–270, Nov. 2017.
8. R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTEXTS.COM, 2014.
9. Jian Zheng, Cencen Xu, Ziang Zhang, and Xiaohua Li, "Electric load forecasting in smart grids using Long-Short-Term-Memory based Recurrent Neural Network," in *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, 2017, pp. 1–6.
10. M. R. Müller, "Análise de desempenho da rede neural artificial ARTMAP fuzzy aplicada para previsão multi-step de cargas elétricas em diferentes níveis de agregação," Repositório Institucional UNESP, 2018.
11. A. Yadav, K. Sahu, Mt. Scholar, D. Cse, and D. Asst Professor, "WIND FORECASTING USING ARTIFICIAL NEURAL NETWORKS: A SURVEY AND TAXONOMY," *Int. J. Res. Sci. Eng.*, vol. 3, no. 2, pp. 148–155, 2017.
12. O. Akinsete and A. Omotosho, "Modeling of Two-Phase Gas Deviation Factor for Gas-Condensate Reservoir using Artificial Neural Network," *Adv. Res.*, vol. 14, no. 1, pp. 1–8, Mar. 2018.
13. O. Falode, C. Udomboso, and F. Ebere, "Prediction of Oilfield Scale formation Using Artificial Neural Network (ANN)," *Adv. Res.*, vol. 7, no. 6, pp. 1–13, Jan. 2016.
14. M. Mazloom, "Application of Neural Networks for Predicting the Workability of Self-Compacting Concrete," *J. Sci. Res. Reports*, vol. 2, no. 1, pp. 429–442, Jan. 2013.
15. M. Benzaghta, "Estimation of Evaporation from a Reservoir in Semi arid Environments Using Artificial Neural Network and Climate Based Models," *Br. J. Appl. Sci. Technol.*, vol. 4, no. 24, pp. 3501–3518, Jan. 2014.

- 412 16. D. Bustan and H. Moodi, "Improving Modelling Accuracy of Aerodynamic Curve of a Wind
413 Turbine Using Neural Networks," *J. Sci. Res. Reports*, vol. 16, no. 3, pp. 1–9, Jan. 2017.
- 414 17. S. Rasouli, H. Tabesh, and K. Etminani, "A Study of Input Variable Selection to Artificial Neural
415 Network for Predicting Hospital Inpatient Flows," *Br. J. Appl. Sci. Technol.*, vol. 18, no. 4, pp.
416 1–8, Jan. 2016.
- 417 18. H. Ghamarnia and Z. Jalili, "Artificial Network for Predicting Water Uptake under Shallow
418 Saline Ground Water Conditions," *J. Sci. Res. Reports*, vol. 7, no. 5, pp. 359–372, Jan. 2015.
- 419 19. M. Patil, D. Lal, S. Karwariya, R. Bhattacharya, and N. Behera, "Comparative Study of
420 Different GCM Models for Stream Flow Prediction," *Curr. J. Appl. Sci. Technol.*, vol. 26, no. 5,
421 pp. 1–12, Apr. 2018.
- 422 20. A. Okon and D. Appah, "Neural Network Models for Predicting Wellhead Pressure-Flow Rate
423 Relationship for Niger Delta Oil Wells," *J. Sci. Res. Reports*, vol. 12, no. 1, pp. 1–14, Jan.
424 2016.
- 425 21. C. Deb, L. S. Eang, J. Yang, and M. Santamouris, "Forecasting diurnal cooling energy load for
426 institutional buildings using Artificial Neural Networks," *Energy Build.*, vol. 121, pp. 284–297,
427 Jun. 2016.
- 428 22. C. Bergmeir, R. J. Hyndman, and J. M. Benítez, "Bagging exponential smoothing methods
429 using STL decomposition and Box–Cox transformation," *Int. J. Forecast.*, vol. 32, no. 2, pp.
430 303–312, Apr. 2016.
- 431 23. G. E. P. Box and D. R. Cox, "An Analysis of Transformations," *Journal of the Royal Statistical
432 Society. Series B (Methodological)*, vol. 26. Wiley Royal Statistical Society, pp. 211–252, 1964.
- 433 24. Pallab Kumar Datta, "An artificial neural network approach for short-term wind speed forecast,"
434 KANSAS STATE UNIVERSITY, 2018.
- 435 25. "R Core Team (2018). R: A language and environment for statistical computing. R Foundation
436 for Statistical Computing, Vienna, Austria." .
- 437 26. C. Bergmeir, R. J. Hyndman, and B. Koo, "A note on the validity of cross-validation for
438 evaluating autoregressive time series prediction," *Comput. Stat. Data Anal.*, vol. 120, pp. 70–
439 83, Apr. 2018.
- 440 27. C. Bergmeir, R. J. Hyndman, and J. M. Benítez, "Bagging Exponential Smoothing Methods
441 using STL Decomposition and Box-Cox Transformation," *Int. J. Forecast.*, 2014.
- 442 28. and I. T. Robert B. Cleveland, William S. Cleveland, Jean E. McRae, "STL: A Seasonal-
443 Decomposition Procedure Based on Loess," *J. Off. Stat.*, vol. 6. No. 1, pp. 3–73, 1990.