Short-term Load Forecasting using Combined Data from Several Weather Stations

Guilherme Guilhermino Neto[1], Henrique S. Hippert[2]

[1]Department of Business Management Technical Course, Federal Institute of Espírito Santo (IFES), Brazil
[2]Department of Statistics, Federal University of Juiz de Fora (UFJF), Brazil

Abstract**—***In order to schedule the load generation and distribution, operators of energy markets rely on short-term load forecasts, especially those made for the next few hours. Since it is not feasible to store a large energy amount for compensating unbalances between supply and demand, what lacks or remains must be exchanged with an interconnected system, at the latest time price quotation. One of the new interests in this research field is the hierarchical load forecasting. The latest smart grid systems made possible to monitor real-time load at various levels of aggregation, from households to the whole system, which brought interest to forecasting from the whole system to a sole household. Some levels may comprise large geographical zones, on which more than one weather station may be located, and that raises a question: how to combine data from more than one weather station, and use the combination as input for load forecasting models? On this paper, we combine data from several weather stations by giving more weight to those stations closer to the centroid of the load zone. We experiment on data from a load zone in the state of New York and 11 weather stations spread throughout the state, using the combined data as input for neural networks. In our datasets, the proposed combinations lead to better results than those from neural networks that use of any of the 11 stations individually. Also, the proposed method outperforms several statistical time series benchmarks.*

Keywords**—hierachical load forecasting, load forecasting, neural networks, time series, weather variables combination.***

I. INTRODUCTION

For many years, electrical energy markets have been structured as vertically integrated monopolies, where a single agent (usually the local government) was responsible for all generation, transmission, and distribution operations. However, in the decade of the 1980s, a worldwide process of liberalization started on these markets, with the goals of promoting efficiency gains, stimulating technical innovation and leading to efficient investment [1].

One of the most important features of today's liberalized markets is that the energy prices are determined by a formal quotation mechanism, strongly influenced by the balance between supply and demand. Because of today's technological standards, it is not feasible to store large amounts of electrical load for compensating unbalance, therefore supply and demand must be matched by selling or buying the difference [1].

Transactions of energy are made on day-ahead markets, with hourly-based prices, where clients buy or sell, in closed auctions, the amount needed for the next 24 hours [1,2]. For this reason, hourly-based day-ahead load forecasts provide vital information for all the stakeholders of energy markets, from the system operators, who must schedule from generation to distribution, to the shareholders, whose bidding also influences the prices [1]. Hourly-based load forecasts made for the next 24 hours, often called short-term load forecasts [3] are the subject of this work.

The short-term behavior of electrical load is known to be seasonal and influenced by the weather. Usually, loads are higher during business hours within a day, and during business days within a week**,** patterns which are periodically repeated. Also, loads are highly correlated to weather variables, mostly the air temperature (for example, higher temperatures usually lead to the use of air-conditioning systems, raising the load levels).

Load forecasting has been a subject of great debate in the literature for decades. The first book solely devoted to the theme is a collection of papers describing methods used by the industry [4]. According to this volume, the practitioner's toolbox at that time consisted mostly of statistical and numerical methods, such as linear regression, exponential smoothing, polynomial curve fitting and trigonometric series. These techniques have the advantages of robustness and interpretability. However, the functional form of models must be pre-defined, which can make the inclusion of the nonlinear relationships between weather and load a challenge.

Statistical and numerical methods are still used by the industry. For example, [5] describes a Box & Jenkins autoregressive model, currently used by Spanish Transmission System Operator.

The evolution of computer hardware made possible to try more computationally demanding methods, such as neural networks, support-vector machines, neural-fuzzy systems, and biological inspired metaheuristics. Because they make possible to capture more specific relationships between load and weather with no need to pre-define a functional form, such methods may reduce the average forecasting errors. However, since these relationships may be too complex, some caution must be taken to avoid overfitting or, equivalently, poor generalization for unseen data.

A broad review and critique on feedforward neural networks, still worth reading, may be found in [6]. More recently, many experiments are being reported using deep learning neural networks (which are even more computationally demanding), such as [7], whichuses pooling deep recurrent networks for household data, and [8], whichuses convolutional neural networks for Greek electrical systems load data.

There does not seem to exist a ‘best’ technique for load forecasting. Let us look for instance at the best results for 2012 edition of Global Energy Competition (GEFCom). GEFCom is an initiative of the Institute of Electrical and Electronics Engineers (IEEE) that invites participants from all over the world to work on solutions for energy forecasting problems. Amongst the best ranked solutions for short-term load forecasting in 2012 were linear regression, polynomial regression, gradient boosting, and neural networks [9]. This reinforces the idea that various techniques may lead to suitable results, and good predictions may depend more on the practitioner’s expertise and its adequacy to the data than on the technique itself.

Despite the wide literature on how to produce load forecasts, new challenges still emerge because of the dynamic nature of energy markets. Some posed problems for the future of energy markets are, according to [2]

1. The impact on climate change on electricity demand, supply and price
2. Share, costs and subsidies to renewables in electricity production
3. The effort to make the electricity sector “smarter”, at different levels of the Electricity Supply Chain, mainly distribution and usage.

All these themes are somehow related to short-term load forecasting and so motivate research on the improvement of existing techniques, the creation of new ones, the solution of new issues, and so on.

One of the main goals of the substitution of old systems by smart grids, mentioned in item (3) of the list above, is the possibility of monitoring the load at various levels of the grid, such as individual transformers, substations, cities, regions and countries. From that follows the need of forecasting not just the load for all the system anymore, but for all individual levels, and for aggregations of them; a paradigm called “hierarchical load forecasting” [10]. As the grid usually covers a vast geographical area, it is possible that, for some zones, data from more than oneweather stations will be available. When that is the case, how can we select the sources of data and combine them?

Little attention has been devoted to strategies for weather station selection until now. The most relevant works are [11] and [12]. The first introduced a heuristic for ranking Nweather stations and, then, combining the best 1 to N by using a simple average. The latter tested the same heuristics but experimented various types of weighted averages for combining the best stations.

Both papers are valuable contributions, since they brought to the table an important practical matter that was overlooked until now. However, there is still much to be developed. For example, testing the heuristic with other forecasting models (both papers only tried linear regression), incorporating to the combining process practical aspects such the localization of the weather stations and the characteristics of the weather data, and trying the proposed selection heuristic on different data.

In this study, we continue the investigations by introducing three new methods for combining weather stations: averages weighted by the distance of the weather station to the load zone, by the difference of the altitudes between the weather station and the load zone and by the intensity of the nonlinear correlation between the weather station and the load datasets. We experiment on temperature data from 11 weather stations in the state of New York, and load data from a zone of the New York Independent System Operator (NYISO).

II. MATERIAL

The data we used for our experiments consist of one series of electrical loads (in MW) and 11 series of dry bulb air temperature (inºC). The measurements correspond to the period from January 1, 2015 to December 31, 2018, adding up to 35,064 hourly observations. The load series, available on the NYISO's website [13], refers to a load zone of the state of New York called GENESE, which comprises the region of Rochester and surroundings. The temperature series are measurements of the actual temperature, reported from 11 weather stations spread through the state of New York. These data are available on the National Oceanic and Atmospheric Administration's (NOAA) website [14]. In Fig. 1, we see the geographical locations of the load zone and the weather stations:

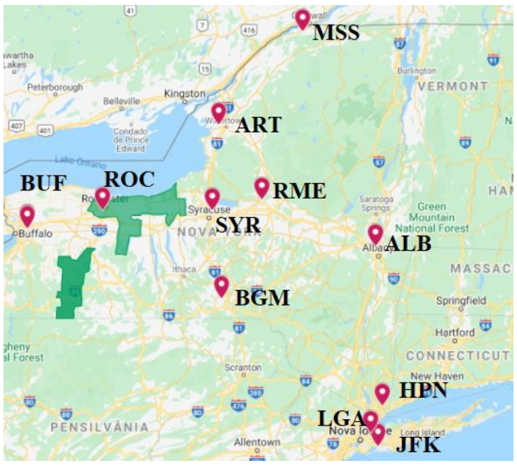

Fig. 1: Borders for the load zone GENESE (highlighted), and locations for 11 weather stations [15].

Despite the GENESE load zone being formed by two disjoint regions, the NYISO treats it as one, and records the loads in a single data set, which accounts for the total load for both areas.

The 11 weather stations, all located in airports, are labeled on the map by their International Air Transport Association (IATA) codes. We will use these codes to refer to the stations from now on. The NOAA provides data from 4 more weather stations (ELM, MSV, POU, SWF). We chose not to use them because their data contained a significant amount of missing values.

The two main characteristics of load series can be observed in our data: the seasonality and the influence of the weather.

In Fig. 2, we have a line plot for two typical fortnights of winter and summer loads, sampled by convenience.

The line plot suggests that two seasonal patterns occur: an intraday pattern, with higher loads in the middle of the day; and an intraweek pattern, with lower loads on the weekends. The plotted data also indicates that the mean level of the load is higher in the winter than in the summer.

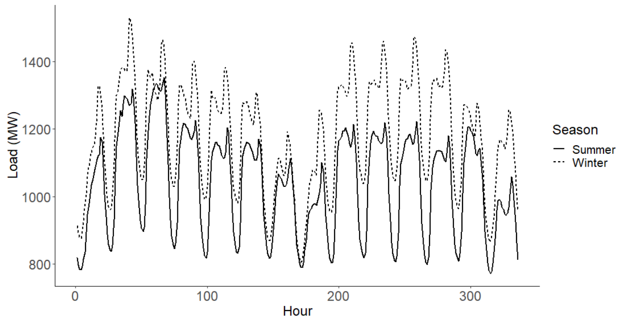

Fig. 2: Typical loads for two fortnights in summer and winter, data for GENESE load zone; data from[13].

In Fig. 3, the scatterplots show the relationship between load and temperature data. For the temperature, we picked the ALB station for convenience, but the pattern is quite similar for all weather stations.

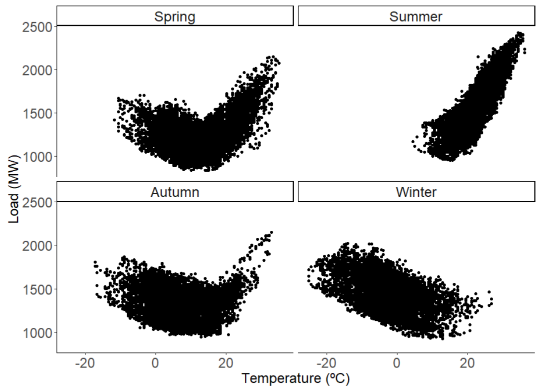

Fig. 3: Scatterplots for loads versus temperatures, by season of the year, for GENESE load zone and ALB weather station; data from[13, 14].

The plots suggest that, for our data, load and temperature are correlated. The highest loads generally occur as we approach the extreme temperatures, what probably happens because of the use of air heating and cooling systems in these cases. In each season, it seems there is a positive linear relationship for summer loads (the higher the temperature, the higher the load) and a negative linear correlation for winter loads (the higher the temperature, the lower the load), while a seemingly nonlinear correlation may be noted for spring and autumn loads (higher loads both associated with lower and higher temperatures).

The patterns for spring and autumn look like a composition of the two other seasons, which we think is due to these seasons being a sort of a transition between the other two.

Almost no treatment was needed for the data. Just a few missing values for the temperature needed to be replaced by imputed values, which we have done by using polynomial interpolation.

III. METHODS

For all the following methods, we will denote the actual and forecasted loads respectively by y_t and $\hat{y}_t$, where t is the instant of time. For our work, we are interested in computing, at the last hour of a day, the forecasts$\hat{y}_{t+k}$, for $k = 1, \ldots 24$. In other words, at the 24th hour of a day, we forecast all the hourly loads for the following day.

3.1SELECTION AND COMBINATION OF WEATHER STATIONS

3.1.1A heuristic for selection and combination

To rank and select the weather stations, we use a heuristic introduced in [11]. Let the temperature series of the$i$-th weather station be denoted by W_i, and n be the number of weather stations.The heuristic follows the five steps enlisted in Table 1:

Table 1. Heuristics for ranking and selecting weather-stations [11, adapted]

#	**Step description**
1	For $i = 1, \ldots, N$ produce in-sample forecasts using the series W_i. Also, at each step, calculate an evaluation measure (the mean absolute percentage error, for example) for these forecasts.
2	Rank the error measures calculated in the previous step in ascending order.
3	For $k = 1, \ldots, n$, generate an artificial weather station by combining the temperature series from the first k stations (which can be done, for example, by using a simple average), and produce in-sample forecasts. At each step, calculate an evaluation measure for these forecasts.
4	Rank the error measures calculated in the previous step in ascending order.
5	The k corresponding to the first position in the previous step's rank will be number of stations to be combined. The first k stations in step 2's ranking will be the ones to be combined.

We have made one little adjustment to this heuristic. In both steps #1 and #3, the forecasts are produced for the in-sample data set. We think that, to assure better generalization, the evaluation for step 3 could be for the post-sample data – and that is how we proceeded in our work.

3.1.2A novel method for combining weather stations

[11] uses just a simple average for combining the weather stations. In [12], weighted averages are tested. Although we think it is reasonable to assign different weights to different stations, the methods proposed in [12] are purely algebraic and mainly based in the forecast errors (more weight is given to the stations that produce more accurate forecasts), leaving out practical issues, such as the geographical location of the station. In this work we introduce a novel method for combining weather stations, based in geographic information.

Bed_i the distance from the weather station i to the centroid of the load zone. The combination of the first k weather stations (step #3 from Table 1) will be given by Eq. (1):

$$W_k = \sum_{i=1}^{k} \frac{d_i^{-1} W_i}{d_i^{-1}} \qquad (1)$$

For our method, the closer the station from the centroid of the load zone, the higher its weight.

The distance between the weather station location and the centroid of the load zone may be calculated by using the haversine formula [16], which results in the great-circle distance between two points on a sphere, given their longitudes and latitudes.

Being $\varphi = (\varphi_1, \varphi_2)$ and $\lambda = (\lambda_1, \lambda_2)$, respectively, the latitude and longitude coordinates, $\Delta\varphi = \varphi_2 - \varphi_1$ and $\Delta\lambda = \lambda_2 - \lambda_1$, and R the radius of the Earth (approximately 6371 km), the distance d between two points on a sphere by the haversine formula is given by the set of equations from Eq. (2.1) to Eq. (2.3):

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2) \qquad (2.1)$$

$$c = 2 \cdot \operatorname{atan2}\left(\sqrt{a}, \sqrt{(1-a)}\right) \qquad (2.2)$$

$$d = R \cdot c \qquad (2.3)$$

3.2 FORECASTING METHODS

In the following sections we describe the forecasting method we propose and the ones we use for benchmarking.

3.2.1Feed Forward Neural Networks (FFNN)

Feed forward neural networks (FFNN) arethe main forecasting method in the work. Thefollowing introduction is based in [17], which the reader may look for more details.

We can define a neural network as a function composition like in Eq. (3):

$$y = f_{NN}(\mathbf{x})\left(f_3\left(\boldsymbol{f}_2(\boldsymbol{f}_1(\mathbf{x}))\right)\right) \quad (3)$$

Eq. (3) is an example of a three-layer neural network. $\boldsymbol{f}_2$ and $\boldsymbol{f}_1$ are vector functions of the form shown in Eq. (4).

$$\boldsymbol{f}_l(\mathbf{z}) \stackrel{\text{def}}{=} \boldsymbol{g}_l(\mathbf{A}_l\mathbf{z} + \mathbf{b}_l) \quad (4)$$

In Eq. (4), l is the index for the layer of the network, an integer greater than or equal to 1. The function $\boldsymbol{g}_l$, usually nonlinear, is called activation function. The values $\mathbf{A}_l$and $\mathbf{b}_l$are respectively called weights and biases of the layer$l$;these values are calculated (or "learned", as in neural networks' practitioner's language) based on the data. The learning process occurs byusing a proper optimization algorithm.

Fig. 4 shows an example ofa neural network in the form of a directed graph, a common visual representation:

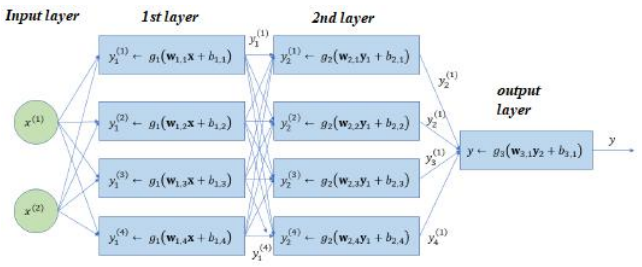
Input layer
1st layer
2nd layer
output layer
$x^{(1)}$
$x^{(2)}$
$y_1^{(1)} \leftarrow g_1(\mathbf{w}_{1,1}\mathbf{x} + b_{1,1})$
$y_1^{(2)} \leftarrow g_1(\mathbf{w}_{1,2}\mathbf{x} + b_{1,2})$
$y_1^{(3)} \leftarrow g_1(\mathbf{w}_{1,3}\mathbf{x} + b_{1,3})$
$y_1^{(4)} \leftarrow g_1(\mathbf{w}_{1,4}\mathbf{x} + b_{1,4})$
$y_1^{(1)}$
$y_1^{(4)}$
$y_2^{(1)} \leftarrow g_2(\mathbf{w}_{2,1}\mathbf{y}_1 + b_{2,1})$
$y_2^{(2)} \leftarrow g_2(\mathbf{w}_{2,2}\mathbf{y}_1 + b_{2,2})$
$y_2^{(3)} \leftarrow g_2(\mathbf{w}_{2,3}\mathbf{y}_1 + b_{2,3})$
$y_2^{(4)} \leftarrow g_2(\mathbf{w}_{2,4}\mathbf{y}_1 + b_{2,4})$
$y_2^{(1)}$
$y_2^{(1)}$
$y_3^{(1)}$
$y_4^{(1)}$
$y \leftarrow g_3(\mathbf{w}_{3,1}\mathbf{y}_2 + b_{3,1})$
y

Fig. 4: FFNN with two inputs, two hidden layers and one output. Adapted from [17].

The architecture depicted in Fig. 4is called feed forward, in the sense that information flows exclusively one-way (in this illustration, from left to right). The rectangles represent the units called neurons, which are organized in layers. In this example, there is an input layer, which receives a dimensional vector$\mathbf{x}$;two hidden (*i.e*. intermediate) layers that perform mathematical operations on the inputs; and an output layer, which returns a value$y$as the result of the previous calculations.

By using the universal approximation theorem, [18] shows that a FFNN with one hidden layer and a finite number of neurons is capable of approximating continuous functions, since the activation functions $\boldsymbol{g}_l$follows some prerequisites, such as being differentiable. This capability makes FFNN attractive for load forecasting because the networks canmodel the complex nonlinear relationship between load and temperature with noneed to specify a functional form.

To produce our forecasts, we propose a FFNN with one hidden layer and the following groups of inputs: 24 loads for day d; 24 loads for day $d-1$; 7 dummy variables for the days of the week; 4 dummy variable for the seasons of the year; 24 temperatures for day d; and 24 temperatures for day $d-1$; 24 temperatures for day $d+1$.

The first three groups model the recent trend of the load curve and, the following two, the daily and weekly seasonal factors. The last three groups model the effect of the temperature on the load.

One may note that although we use the temperatures of the day $d+1$ in our model, we do not have access to these data in practice: in fact, because the forecasts are made at the day d, the temperature series for the next day does not even exists at this point. In practice we would use temperature forecasts, instead of the actual temperatures. Since temperature forecasts contain errors, we also made a sensitivity analysis by injecting various levels of noise to the temperature data to see how the errors in temperature data would affect the performance of our model.

Another thing that is important to highlight is how we proceeded for the first phase of the implementation of the FFNN: the hyperparameter tuning.

We adopted the strategy suggested by [19]: start by deciding which hyper parameters we are willing to tune (which may be limited in number, according to time and computational power issues) and fix the values for the others. For all the hyper parameters that will be tuned, from the most important to the least, try a number (defined by the user) of random values for the parameter while fixing the others. Pick the value that leads to the best results, make it fixed and go to the next hyper parameter. [19] points that this random search has proven more effective than a grid search for hyper parameter tuning. In the order of importance, we have chosen to tune the following hyper parameters: learning rate, batch size, number of hidden units and number of iterations. Some other hyper parameters (and their fixed values) were: number of hidden layers (1), learning algorithm (Adam) and activation function (ReLU).

A common practice when tuning the hyper parameters of a FFNN in a limited data sample is using a resample technique such as cross-validation, with the goal to assure better generalization. One of the most common forms of cross-validation is the k-fold cross validation. For this method, data is first randomly shuffled, then split in k groups. For each group, its data is taken as an out-of-sample (or validation) set while the rest is taken as an in-sample set. The model is fit using the in-sample data and evaluated for the out-of-sample data. The final evaluation is done by summarizing the performance for all k groups.

We must be careful, though, if there is time dependence**:** shuffling the data may cause us to use future data for fitting the model and past data for evaluating. Since usingthe future to predict the past would seem rather unfair, we should look for a more suitable strategy.

We used the strategy usually called “rolling forecasting origin”. In this approach, the validation sets all consist of one observation; the related training sets will be all the observations that occurred before [20].

Fig. 5 illustrates how this procedure works: the blue dots represent the training sets and the red dots the validation sets. The diagram makes clears that no future observation is used during the training.

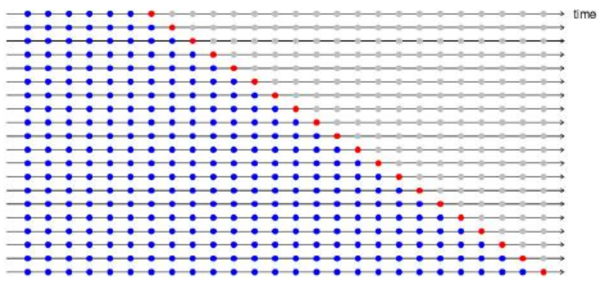

Fig. 5: Diagram for the general rolling forecasting origin method [20]

Although our data is limited, we still have some thousands of points. Because of that, we defined the validation sets to comprise a month of observations, instead of a single observation. Also, for each new validation set, we included six more months instead of only one - what we think that makes more sense, since our data is seasonal.

The final segmentation we used for validating the FFNN models is shown in Table 4.

Table 3. Adapted rolling forecasting origin segmentation

Set	In-sample months	Validation month
1	[1,5]	6
2	[1,11]	12
3	[1,17]	18
4	[1,23]	24
5	[1,29]	30
6	[1,35]	36
7	[1,41]	42
8	[1,47]	48

3.2.2Benchmark I: Seasonal naïve method

A naïve method, possible the simplest class of forecasting methods, considers that the forecasting equals the last observed value.

Our data is seasonal, so it is reasonable to use a variation called seasonal naïve [21]. The formulation can be read in the Eq. (6):

$$\hat{y}_t = y_{t-m} \qquad (6)$$

In Eq. (6), m is the seasonal period. As the data is hourly and we have two seasonal periods, with $m = 24$ hours (intraday) and $m = 168$hours (intraweek), we experimented two seasonal naïve forecasters.

3.3.3Benchmark II: Double-seasonal exponential smoothing

A more sophisticated time series method is the extension of the classical Holt-Winters exponential smoothing, proposed by [22] to accommodate two seasonal factors. The double-seasonal exponential smoothing (also called Holt-Winters-Taylor exponential smoothing, or HWT) is given by the set Eq. (7.1) to (7.4):

$$l_t = \alpha\left(y_t - d_{t-s_1} - w_{t-s_2}\right) + (1-\alpha)l_{t-1} \qquad (7.1)$$

$$d_t = \delta\left(y_t - l_t - w_{t-s_2}\right) + (1-\delta)d_{t-s_1} \qquad (7.2)$$

$$w_t = \omega\left(y_t - l_t - d_{t-s_1}\right) + (1-\omega)w_{t-s_2} \qquad (7.3)$$

$$\hat{y}_{t+k} = l_t + d_{t-s_1+k} + w_{t-s_2+k} + \phi^k\left(y_t - \left(l_{t-1} + d_{t-s_1} + w_{t-s_2}\right)\right) \qquad (7.4)$$

In the Eq. (7.1) to (7.4), l_t is the level;d_t is the intraday seasonal factor; andw_t is the intraweek seasonal factor, for instant t. The forecast $\hat{y}_{t+k}$is a composition of the most recent updates of these components and a correction of the last forecast. α, δ, and ω are constants for level, intraday seasonal factor, and intraweek seasonal factor. ϕ is a constant for correcting the first order autocorrelation. The constants are real numbers, restricted to the interval [0,1].

Although relatively simple, this method has been proved surprisingly accurate for different time series, even shown to outperform more complex multivariate methods, as reported in [23, 24].

3.3.3 Benchmark III: Box & Jenkins models

A last benchmark we have tried for our data was the class of models called Box & Jenkins models [25], which are an auto-regressive formulation for time series.

An (p, q) order auto-regressive (AR) moving average (MA) polynomial, denoted by $ARMA(p, q)$, is written as in Eq. (8):

$$\phi(B)y_t = \theta(B)\varepsilon_t \qquad (8)$$

,where $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p$; $\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \ldots - \theta_q B^q$; B^h is the backshift operator, which lags and observation by h steps (that is, $B^h y_t = y_{t-h}$); $\varepsilon_t \sim N(0, \sigma^2)$; and ϕ andθ are the model's coefficients.

We can derive a seasonal formulation for this model. Denoted by$ARMA(P,Q)_s$, the (P,Q) order seasonal formulation with period s, is written as in Eq. (9):

$$\Phi(B^s)y_t = \Theta(B^s)\varepsilon_t \qquad (9)$$

, where $\Phi(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - \ldots - \Phi_P B^{Ps}$; $\Theta(B^s) = 1 - \Theta_1 B^s - \Theta_2 B^{2s} - \ldots - \Theta_Q B^{Qs}$; and Φ andΘ are constants.

Finally, a mixed seasonal ARMA, denoted by $ARMA(p,q) \times ARMA(P,Q)_s$, a composition of both aforementioned models, is shown in Eq. (10):

$$\Phi(B^s)\phi(B)y_t = \Theta(B^s)\theta(B)\varepsilon_t \qquad (10)$$

The formulations in Eq. (8) to (10) do not directly allow us to forecast for lead times ahead of $t+1$. To achieve our goal of forecasting for up to 24 hours ahead, we have adjusted 24 Box & Jenkins models – one for each hour of the day and used them to forecast the next hour.

3.3 EVALUATION

For evaluating the post-sample accuracy, we chose to calculate the mean absolute percentage error (MAPE), as it is a standard in the literature, and is easily interpretable.

The MAPE is given by the following equation:

$$\text{MAPE} = \frac{1}{N}\sum_{1}^{N}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \times 100 \qquad (11)$$

In Eq. (11), N is the number of observations in the data set.

3.4 COMPUTATIONAL RESOURCES

All the experiments were run on a 3.70GHz Intel® Core™ i7-8700K CPU, with 32 GB of RAM, and a Nvidia GeForce 1080Ti GPU with 11 MB RAM.

The implementations were written in R language [26]. We have used functions from the packages keras [27], for neural networks;forecast [28] for Box and Jenkins models; and geosphere [29] for haversine distance.

For calculating the centroid of the load zone, we have used the coordinates of the outline map made available by [30,31].

IV. RESULTS AND DISCUSSION

We used approximately 36 months, up to 31 December, 2017, for our in-sample data. These data were used for optimizing the HWT constants, finding the Box & Jenkins models and training the FFNNs.

The data for the final year, 2018, we used for post-sample evaluation.

4.1 BENCHMARKS

First, we implemented the time series benchmark methods. We used the limited memory Broyden-Fletcher-Goldfarb-Shano (l-BFGS) optimization algorithm [32] to find the constants that minimized the in-sample mean squared error. To find the order and the model's coefficients for the Box & Jenkins models, we used the Hyndman-Kandakhar algorithm [33], which minimizes the Akaike Information Criterion (AIC), leading to more parsimonious models.

Table 2 shows the MAPE for the benchmark methods, for out-of-sample data.

Table 2. MAPE for the benchmark methods (out-of-sample data)

Method	MAPE (%)
HWT	4.15
Box & Jenkins	5.31
Seasonal naïve ($m = 24$)	6.64
Seasonal naïve ($m = 168$)	8.75

As we expected, the naïve forecasters perform worse than the other two methods. HWT produces more accurate forecasts than Box & Jenkins models, according to the MAPE, and is the best of the benchmarks.

In Table 3 we show the constants for the model. While the constant for the level update,λ, is close to zero, which suggests that the average level is nearly constant for all the framed time period, the constant for adjustment of the first order autocorrelation, ϕ, pays a great role in our data. As regards the intraday and intraweek seasonal factors, their constants, δ and ω indicate that both influence the forecasts with similar strength.

Table 3. Constants calculated for HWT method

ϕ	λ	δ	ω
0.99	0.02	0.14	0.15

4.2 FFNN

4.2.1 One weather station per FFNN

We started by training 11 FFNNs, all with architecture mentioned in section 3.2.1. That is: the inputs were 24 loads for day d; 24 loads for day d-1; 7 dummy variables for the days of the week; 4 dummy variable for the seasons of the year; 24 temperatures for day d; and 24 temperatures for day d-1; 24 temperatures for day d+1; and the outputs were the 24 loads for day d+1.

As we mentioned before, although we have the data for observed temperatures for day d+1, in practice we have just forecasts for those. Because we aim at testing our method for a broad range of situations, we also evaluated the networks with noise added to the data, to simulate the temperature forecast errors. The level of noise r added to a temperature data point in time instant t is shown in Eq. (12).

$$\mathrm{r}_t = \mathrm{rnorm}(0, l \times \mu_{TMP}) \qquad (12)$$

In Eq. (12), the noise is given by a random number of the gaussian distribution (rnorm), with 0 mean and a standard deviation given by a constant l times the mean of the temperature series.

In short, for each weather station data set, we:

1. Train 100 FFNNs (because initial random guesses in the training phase may cause the results to vary, and running more networks we can analyze the variability)
2. Add noise to the out-of-sample data
3. Calculate the median MAPE for the predictions of the 100 FFNNs in the out-of-sample data.

We have tried this algorithm for noise levels l = 1%, 2%, 3%, 4% and 5%, and also for data with no noise (l = 0).

The final evaluation of the post-sample accuracy was given by the median MAPE for the performance of the FFNN in all levels of noise – which we think coversthe wide range of possibilities that may come from the variability regarded to the initial guesses in the training phase, and also from the noise in the temperature data.

Table 4 show the results for 11 FFNNs, each using data from a different weather station.We can also see the distance from each weather station to the centroid of the load zone. We mentioned in section II that GENESE load zone is formed by to disjoint geographical areas. Among other possible strategies, we calculated this distance as the average of the distance to the centroids of the two areas.

Table 4.MAPEs for FFNNs (different stations)

Weather station	**MAPE (%)**	**distance (10^5 m)**
ROC	3.32	0.65
SYR	3.39	1.33
BUF	3.41	1.06
ART	3.63	1.91
BGM	3.68	1.56
RME	3.79	1.91
MSS	3.87	3.31
ALB	3.99	3.15
LGA	4.53	3.84
JFK	4.90	4.00
HPN	4.94	3.78

Table 4 seems to indicate that, the closer the station is to the centroid, the best are the FFNN results. That is made clearer by observing the scatterplot for MAPE versus distance in Fig. 6:

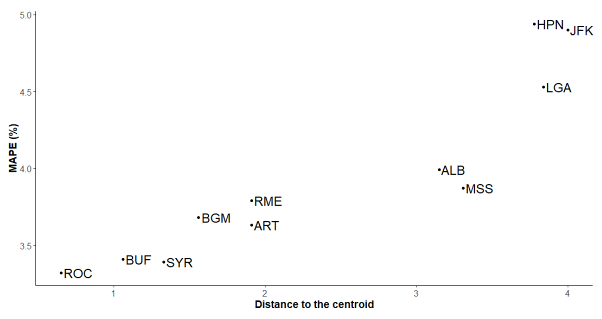

Fig. 6: Scatterplot for MAPE versus distance, weather stations labeled (FFNN)

The points in Fig. 6 have a Pearson's correlation coefficient of 0.91, which suggests a strong positive linear correlation (the higher the distance between the weather station and the centroid, the higher the MAPE).

We then started to combine the weather stations, from two station to 11, starting from the best (following the heuristics mentioned in 3.1), re-trained the FFNNs and computed the MAPEs for the various levels of noises.

The first combination we experimented was a simple average of the temperature data. Results are displayed in Table 5. The simple average improves the results we had previously, especially the first three stations from Table 4, ROC, SYR and BUF, which are the closest to the load zone.

Table 5. MAPEs for FFNNs (combination by mean)

No. of weather stations	MAPE (%)
3	3.09
5	3.11
4	3.12
2	3.14
7	3.18
6	3.19
8	3.25
10	3.29
1	3.32
9	3.33
11	3.44

Finally, we experimented our proposed combination method, the weighted average by the inverse of the distance to the centroid. Results shown in Table 6point that our method improves the results even more than the combination by simple average: combining only the 2 best weather stations of the ranking in Table 2 lead to a significantly better performance.

Table 6. MAPEs for FFNNs (weighted average by the inverse of the distance to the centroid of the load zone)

No. of weather stations	MAPE (%)
2	2,92
5	2,92
3	2,95
4	2,96
9	3,04
6	3,05
7	3,05
8	3,06
11	3,10
10	3,12
1	3,32

4.4 A BRIEF SENSITIVITY ANALYSIS

A last analysis we would like to show is a brief sensitivity analysis to the injected noise.

In Table 7, we show the median MAPEs for the best FFNN (stations ROC and SYR, combined by the centroid method), for the various noise levels and also the MAPEs for the benchmark methods:

Table 7. Best methods

	Noise level					
Method	**0**	**1%**	**2%**	**3%**	**4%**	**5%**
FFNN	2.59	2.64	2.78	2.95	3.19	3.48
HWT	4.15					
SARMA	5.31					
NF24	6.64					
NF168	8.75					

Those results are also shown in Fig. 7.

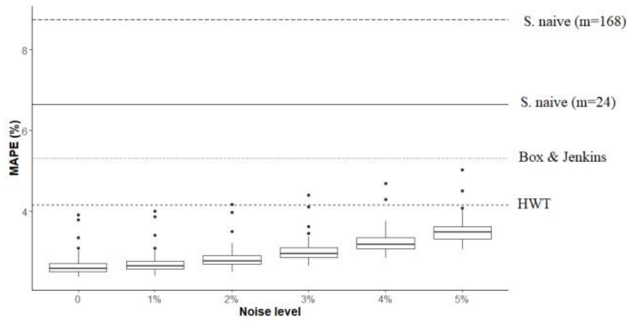

Fig. 7: Results for the best FFNN and the benchmark methods

Note that these results show that, even though the FFNNs are capable of producing accurate forecasts, they are not flawless. Random initial guesses in the training phase may cause variations, as we can see in Fig. 7. Even with no noiseadded, some outlier MAPEs were observed.

But not just that: as we inject more noise to the temperature data, the variability gets higher (which is clear by looking at the size of the boxes) and some few results may even be worse than those produced by an exponential smoothing method, which is much less computationally demanding.

Because of that, we recommend running FFNNs as many times as possible, in order to analyze the variability, and being careful to consider the possibility that the input data may include some noise. If that is the case, it is possible that a simpler, but more robust method, such the time series benchmarks we used, area safer bet, depending on the level of noise.

V. CONCLUSION

With the emerging importance of hierarchical load forecasting, the problem of selecting and combining weather stations data becomes a major concern for the field of load forecasting.

In this work, we propose a new method for combining weather station data, a weighted average where more weight is assigned the closer the weather station is to the load zone. Using data from a load zone of the NYISO and 11 temperature weather stations of the state of New York, we experimented with this method for combining the data, and usedthe combinations as input for neural networks.

For our data, the proposed combining method improves the performance of the neural networks, if compared to the use of the weather stations individually and to the combination via simple averages. The method also outperforms, in median, several time series benchmarks. For certain levels of noise in the input data, though, one must be careful since it may weaken the results at some level, as we commented in a brief sensitivity analysis.

For further work, we suggest trying the method on other datasets, tryingwith different neural network architectures and trying to combine the results from the HWT exponential smoothing with those from the neural networks, in order to see if the robustness of the time series method could improve the performance of the neural networks when there is noise in the input data.

ACKNOWLEDGEMENTS

We would like to thank Federal Institute of Espírito Santo (IFES) – campus Linhares,the postgraduate program on Computational Modelling and the Department of Statistics of the Federal University of Juiz de Fora (UFJF) for supporting this work.

REFERENCES

[1] Weron, R. (2006). *Modeling and forecasting electricity loads and prices: A statistical approach.* Chichester, UK: Wiley.

[2] Cretì, A., &Fontini, F. (2019). *Economics of electricity: markets, competition and rules*. Cambridge, UK: Cambridge University Press.

[3] Hong, T.,& Fan, S. (2016). Probabilistics electric load forecasting: A tutorial review.*International Journal of Forecasting*, 32 (3), 914-938.

[4] Bunn, D.W.,& Farmer, E.D. (1985). *Comparative models for electrical load forecasting.* Belfast, UK: Wiley.

[5] Caro, E., Juan, J.,& Cara, J. (2020). Periodically correlated models for short-term electricity load forecasting.*Applied Mathematics and Computation*, 364, 124642

[6] Hippert, H.S., Pedreira, C.E.,& Souza, R.C. (2001). Neural networks for short-term load forecasting: A review and evaluation.*IEEE Transactions on Power Systems*, 16 (1), 44-55.

[7] Shi, H., Xu, M.,& Li, R. (2018). Deep learning for household load forecasting – A novel pooling deep RNN.*IEEE Transactions on Smart Grid*, 9 (5), 5271-5280.

[8] Sideratos, G., Ikonomopoulos, A.,& Hatziagyriou, N.D. (2020). A novel fuzzy-based ensemble model for load forecasting using hybrid deep neural networks.*Electric Power Systems Research*, 178, 106025.

[9] Hong, T., Pinson, P., &Fan, S. (2014). Global Energy Forecasting Competition 2012.*International Journal of Forecasting*, 30 (2), 357-363.

[10] Ben Taieb, S., Taylor, J.W., &Hyndman, R.J. Hierarchical probabilistic forecasting of electricity demand with smart meter data.*Journal of the American Statistical Association*, to be published.

[11] Hong, T., Wang, P., &White, L. (2015). Weather station selection for electric load forecasting.*International Journal of Forecasting*, 31 (2), 286-295.

[12] Sobhani, M., Campbell, A., Sangamwar, S., Li, C., &Hong, T. (2019). Combining weather stations for electric load forecasting.*Energies*, 12 (8), 1510.

[13] New York Independent System Operator. *Real-time dashboard*. Retrieved September 23, 2020 from:https://www.nyiso.com/real-time-dashboard

[14] National Oceanic and Atmospheric Administration. *Integrated surface dataset (Global)*. Retrieved September 23, 2020 from:https://www.ncei.noaa.gov/access/search/data-search/global-hourly

[15] Google. (n.d.). [Google Maps directions to the State of New York and surroundings]. Retrieved September 23, 2020 from https://goo.gl/maps/DogN3sf4mUCMvLY58

[16] Sinnott, R.W. (1984). Virtues of the haversine. *Sky and Telescope*, 68 (2), 159.

[17] Burkov, A. (2019). Neural Networks and Deep Learning. In*The hundred-page machine learning book* (pp. 61-75). Andriy Burkov.

[18] Cyberenko, G. (1989). Approximations by superpositions of sigmoidal functions. *Mathematics of Control, Signals, and Systems*, 2 (4), 303-314.

[19] Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In: *Neural networks: Tricks of the trade* (pp. 437–478). Berlin: Springer.

[20] Hyndman, R.J. (2016). *Cross-validation for time series.* Retrieved September 23, 2020 from: https://robjhyndman.com/hyndsight/tscv/

[21] Hyndman, R.J., & Athanasopoulos, G. (2018). Some simple forecasting methods. In *Forecasting: principles and practice, 2nd edition*, OTexts: Melbourne, Australia. Retrieved 23 September, 2020 from: OTexts.com/fpp2.

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

[23] Taylor, J.W. (2010). Triple seasonal methods for short-term load forecasting. *European Journal of Operational Research*, 204 (1), 139-152.
[24] Taylor, J.W. (2012). Short-term load forecasting with exponentially weighted methods. *IEEE Transactions on Power Systems*, 27 (1), 458-464.
[25] Box, G.E.P., Jenkins, G.M., Reinsel, G.C., & Ljung, G.M. *Time series analysis: forecasting and control.* 5th ed. Hoboken, US: 2016.
[26] R Core Team (2020). R: A Language and Environment for Statistical Computing. Vienna, Austria. Retrieved from: https://www.R-project.org
[27] Falbel. D., Allaire, J.J., Chollet, F., RStudio, Google, Tang, Y., Van Der Bijl, W., Studer, M., &Keydana, S. (2020). keras: R Interface to ‘Keras’. R package version 2.3.0.0.
[28] Hyndman, R., Athanasopoulos, G., Bergmeir, C., Caceres, G., Chhay, L., O’Hara-Wild, M., Petropoulos, F., &Razbash, S. (2020). forecast: Forecasting functions for time series and linear models. R package version 8.13.
[29] Hijmans, R.J., Williams, E., & Vennes, C. (2019). geosphere: Spherical Trigonometry. R package version 1.5-10.
[30] Igava, L. *Zone B one outline*. Retrieved Sept23rd, 2020 from:https://www.arcgis.com/home/item.html?id=99fa9c4d22274fa7bc813e48d0ee28cd
[31] Igava, L. *Zone B two outline*. Retrieved Sept23rd, 2020 from: https://www.arcgis.com/home/item.html?id=a6054c840e9047d9a648ed89d1c741ea
[32] Byrd, R.H., Lu, P.; Nocedal, J., & Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific and Statistical Computing*, 16 (5), 1190-1208.
[33] Hyndman, R.J., & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, 27 (3), 1-22.