Available online at www.sciencedirect.com
ScienceDirect
Mathematics and Computers in Simulation 224 (2024) 63–75
MATHEMATICS AND COMPUTERS IN SIMULATION
www.elsevier.com/locate/matcom

Original articles

Day-ahead forecasting of residential electric power consumption for energy management using Long Short-Term Memory encoder–decoder model

G. La Tona*, M. Luna, M.C. Di Piazza

Consiglio Nazionale delle Ricerche (CNR), Istituto di Ingegneria del Mare (INM), Palermo, Italy

Received 24 October 2022; received in revised form 16 May 2023; accepted 23 June 2023
Available online 3 July 2023

Abstract

Energy management in smart buildings and energy communities needs short-term load demand forecasting for optimization-based scheduling, dispatch, and real-time operation. However, producing accurate forecasting for individual residential households is more challenging compared to the forecasting of load demand at the distribution level, which is smoother and benefits from statistical compensation of errors. This paper presents a day-ahead forecasting technique for individual residential load demand that is based on the Long Short-Term Memory encoder–decoder architecture, which is extended to consider possibly differing sets of past and future exogenous variables. A novel focus is posed on the validation of the proposed approach considering that it is tailored for use by energy management systems. A publicly available dataset was used for validation, and the approach was compared with three other methods, resulting in a reduction of the Mean Absolute Scaled Error by up to 8%.

Keywords: Forecasting; Residential electrical consumption; Energy management; LSTM

1. Introduction

Energy waste and pollutant emission reduction is nowadays considered a nondeferrable priority. The research community and policy makers have been trying to move towards smarter and more efficient energy use paradigms. To this end, Energy Communities sharing common energy sources [5] and smart buildings enabling efficient use of energy are among the main pillars of this change.

Energy Management Systems (EMSs) play a key enabling role in the implementation of such strategies. They manage power flows in the electrical microgrids among renewable sources, loads, electrical storage devices, and the main grid. Furthermore, EMSs may pursue several goals like the maximization of operational efficiency, the minimization of emissions, or the reduction of sources and load uncertainty [20].

Predicting the behavior of the energy system is crucial to mitigate potential uncertainties. Therefore, accurate forecasting of electric load demand is fundamental for planning and operation management at several scales, from

* Corresponding author at: INM-CNR, via Ugo La Malfa 153, 90146, Palermo, Italy.
E-mail address: giuseppe.latona@cnr.it (G. La Tona).

https://doi.org/10.1016/j.matcom.2023.06.017

transmission and distribution system operators to microgrid or building EMSs. The former are interested in short-term and medium-term aggregate load demand forecasting for day-ahead scheduling, unit commitment, and energy trading. The latter are interested in short-term load forecasting of both aggregated and individual users' load demand for day-ahead scheduling and real-time control [10].

This paper focuses on forecasting individual residential household load demand, i.e., the profile of interest for residential EMSs. Such a profile is more volatile than the aggregated load demand. Therefore, its forecasting is more challenging. In fact, the aggregated load demand has a smoother profile due to the statistical compensation between the many users' profiles [32].

Not many papers have been proposed in the literature about short-term load demand forecasting of individual residential households. In fact, there is a scarcity of datasets compared to aggregated load profiles. However, due to the adoption of smart metering infrastructures and the diffusion of Internet of Things devices, such datasets are starting to become available to researchers.

Recently, pure deep learning methods have started being applied to time series forecasting with good results. These end-to-end data-driven methods do not require expert analysis of the historical data, manual extraction of features, or decomposition into trend and seasonality components. Historically, pure deep learning methods used to perform worse than hybrid methods combining classical statistical approaches and machine learning models in time series forecasting competitions [23]. However, the proposal of new architectures and the availability of bigger datasets have paved the way for new approaches that improved over the state of the art of time series forecasting [27].

Recurrent Neural Networks (RNNs) have been used to forecast individual residential household load demand. A simple RNN is used in [33], whereas Gated Recurrent Units and Long Short-Term Memory (LSTMs) are used in [17], and a LSTM is combined with Fully Connected (FC) layers in [16]. Furthermore, different combinations of Convolutional Neural Networks (CNNs) and RNNs have been explored. CNNs were combined with stacks of LSTMs in [2,3,15], whereas in [21,36] they were combined with Bidirectional LSTMs and in [30,34] with GRUs. Finally, dilated CNNs were combined with Bidirectional LSTMs in [14]. Some of these methods consider exogenous variables (i.e., variables whose values are determined outside the model and affecting the model without being affected by it). However, all these approaches can only be used for fixed-size horizon forecasting and are only compatible with past exogenous variables (i.e., historical values known at the time of forecasting).

The encoder–decoder architecture was proposed in the literature to deal with variable-length sequences. This architecture has been extended to time-series forecasting and in particular the model proposed in [24] used an LSTM encoder–decoder architecture to forecast individual residential household load demand. The model is capable of variable-size horizon forecasting and of considering future exogenous variables. However, the encoder must be pretrained, and the future exogenous and past exogenous variables must coincide. Also, only a linear transformation is applied to decoder output, although it has been shown that the nonlinearities added by final FC layers are beneficial.

Other RNN encoder–decoder models were proposed for aggregate multihousehold-level load forecasting in [28, 31]. These models are only compatible with past exogenous variables because only recurrent outputs are given as inputs to the decoder (initialized with the encoder state). An LSTM encoder–decoder for regional load demand was proposed in [38]. Future exogenous variables are concatenated with the recurrent output of the decoder and are given as an input to the decoder. However, future exogenous variables must coincide with past exogenous ones.

Other approaches not considering exogenous variables, based on recurrence plots and CNNs [29] and on Conditional Restricted Boltzmann Machines [25], have been proposed. Furthermore, learning from multiple houses in a neighborhood for forecasting multiple individual profiles was proposed in [32]. A pooling deep RNN was used to learn shared uncertainties. However, this approach requires an appropriate dataset and sharing of individual load consumption profiles with a centralized forecaster, which may lead to privacy issues.

The use of exogenous variables has proven to be beneficial in residential load demand forecasting. Furthermore, the constraint of using the same sets of past and future exogenous variables is very limiting. Therefore, in this paper a novel way to extend the LSTM encoder–decoder architecture that can consider different sets of past and future exogenous variables is proposed.

Crucial aspects of short-term forecasting for EMSs are also the choice of appropriate error metrics and the definition of a suitable testing strategy. Too often, model performance is measured on the forecasting of just one horizon span (e.g., the forecasting of one day) [14,17,23]. However, this is not a significant sample for forecasting

methods used by EMSs that require new forecasting at least daily without retraining the model. Therefore, it is important to provide performance indices based on multiple repeated forecasting over a sliding horizon of historical data, a procedure known as backtesting without retraining. Furthermore, for reproducibility of the results, it is also important to discuss how to aggregate error metrics over multiple forecasting. The proposed model is tested in this paper using backtesting without retraining, and two different methods for aggregating the error metrics are discussed.

The main contributions of the paper can be summarized as follows:

- A novel way to use the LSTM encoder–decoder architecture with both past and future exogenous inputs for day-ahead individual household power consumption forecasting is proposed.
- The proposed model can produce forecasting with variable-length horizon, it can consider different sets of past and future exogenous variables, and there is no need to pre-train the decoder.
- The proposed model outperforms other benchmark models on a public dataset.
- The validation is performed through backtesting on a test set without retraining the model, and different methods for aggregating error metrics are proposed.

The rest of this paper is structured as follows: Section 2 presents the proposed method and the case study, together with a discussion of how the method is validated and compared to other models; Section 3 presents the results and the relative discussion; Section 4 reports the conclusions, and Appendix A summarizes the supplementary materials of the paper.

2. Materials and method

The proposed approach is discussed in this section after briefly recalling the formal definition of time series and the peculiarities of load demand time series. The LSTM encoder–decoder is introduced, and the way it is extended to time series forecasting is discussed. Finally, the case study is presented, including how the data are prepared and how the method is validated.

2.1. Load demand time series definitions

Time series are defined as sets of data points indexed by time, usually taken at regular time intervals. They can be modeled as stochastic processes, which, in the discrete case, are sequences of random variables. Forecasting means predicting the value of the random variable y_t given the previous observations $(y_{t-1}, \ldots, y_1)$. Furthermore, it is usually desired to forecast the series for a horizon h in the future $(y_{t+h-1}, \ldots, y_t)$. The considered time series can be univariate, if the data points are scalar, or multivariate, if the data points are vectors.

It is often useful to consider other related variables called exogenous variables. Their values can be contemporaneous to the observed values of the considered time series (past exogenous variables) or can also be known for future time steps, contemporaneous to the forecasting horizon (future exogenous variables). An example of future exogenous variables are calendar features, which are known in advance.

In general, therefore, a forecasting model approximates the relationship between future values of the considered time series given previous observations, past exogenous features, and future exogenous features, as in:

$$(y_{t+h-1}, \ldots, y_t) = f(\boldsymbol{z_{t+h-1}}, \ldots, \boldsymbol{z_t}, y_{t-1}, \ldots, y_1, \boldsymbol{x_{t-1}}, \ldots, \boldsymbol{x_1}) \tag{1}$$

where f is the unknown function that determines the evolution of the time series, and $\boldsymbol{z_{t+h-1}}, \ldots, \boldsymbol{z_t}$ and $\boldsymbol{x_{t-1}}, \ldots, \boldsymbol{x_1}$ indicate future exogenous variables and past exogenous variables, respectively [12].

Load demand time series in individual residential households usually consist of aggregated (at the household level) active power demand measurements. Useful exogenous variables are weather measurements like ambient temperature and other electrical measurements, like submetering data. Furthermore, calendar features like time of day, day of the year, and holiday or weekend indicators are exogenous variables that are very useful because correlated with household users' habits and behaviors.

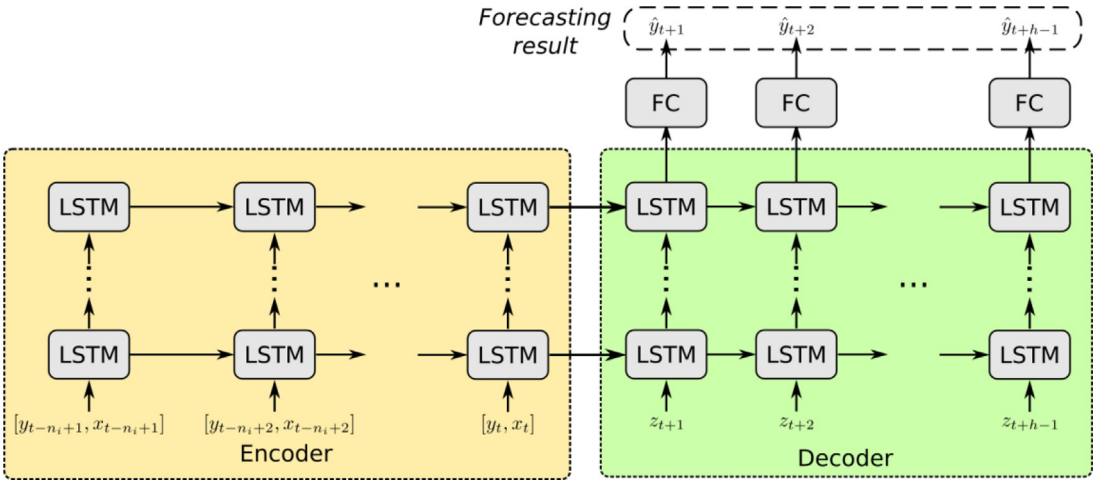

Fig. 1. Proposed model architecture. On the left, the encoder block through LSTM layers receives the past observations and past exogenous inputs. The resulting encoding is the initial state of the LSTM layers of the decoder, which receive as inputs the future exogenous values. The output of the decoder is processed through fully connected layers to produce the resulting forecasting.

2.2. LSTM encoder–decoder with exogenous inputs model

The model proposed in this paper was designed with the following goals: (1) the model must support end-to-end learning (no hand-coded feature extraction), (2) the input window and the output window do not have to be of the same length, (3) variable-length forecasting horizon must be supported, (4) both past and future exogenous variables must be supported and the two sets of variable do not have to coincide.

Recurrent Neural Networks (RNNs) are intrinsically able to work with sequential data. However, when trained over long sequences they suffer from vanishing or exploding gradients. Therefore, RNNs are unable to learn long-term dependencies on data, like what is needed in load demand timeseries, which exhibit multiple seasonality.

The building block of the proposed forecasting model is the LSTM RNN cell. LSTM is an architecture developed with the purpose to overcome vanishing and exploding gradient issues (typical of RNNs) and learn long-term dependencies. An LSTM consists of a memory cell that can maintain its state over time and of three nonlinear gating units (input gate, output gate, and forget gate) that regulate the information flow into and out of the cell [7].

LSTM (like all RNNs in general) by itself is not able to predict variable-length sequences, nor it can easily consider future values of exogenous variables. To address these limitations, the sequence-to-sequence or RNN encoder–decoder architecture was proposed in [35].

The approach proposed in this paper is based on the LSTM RNN encoder–decoder architecture and is summarized in Fig. 1. It consists of a stack of LSTM layers that receive as input a window of past observations of the considered time series and of the exogenous time series, whereas it outputs an encoding of such observations. Another stack of LSTM layers has as initial state the encoding and receives as input the future values of the exogenous time series for the forecasting horizon. Its output is connected to a stack of fully connected (FC) layers that output the predicted values of the considered time series for the forecasting horizon. The network activation function for the FC layers is the Rectified Linear Unit (ReLU) $g(x) = \max\{0, x\}$ (where x is the result of the weighted sum of previous layer outputs), except for the output layer, which has linear activation.

It is worth observing that this architecture is compatible with multivariate input series, and future exogenous variables do not have to coincide with past exogenous variables. Furthermore, in a such pure deep learning fashion (i.e., no manual feature extraction or classical time series decomposition), the network learns the features of the time series at successive layers of abstraction and the time relations between such features, encoding them into a latent representation. The network also learns the relations between future exogenous variables, the encoded state, and the future observed values of the time series. The proposed model can produce variable-length forecasting. In fact, it produces an output sequence that has the same length of the future exogenous variables passed to it.

Table 1
Variables of the IHEPC dataset.

Variable	Description
global_active_power	household global active power (kW)
global_reactive_power	household global reactive power (kW)
voltage	power system voltage (V)
global_intensity	household global current intensity (A)
sub_metering_1	energy sub-metering No. 1: kitchen (Wh)
sub_metering_2	energy sub-metering No. 2: laundry room (Wh)
sub_metering_3	energy sub-metering No. 3: electric water-heater and air-conditioner (Wh)

To increase generalization and avoid over-fitting the model, three complementary regularization strategies have been followed. First, L1 regularization of layer parameters is used. This adds a term $\lambda \sum_{i=0}^{N} |\theta_i|$ (θ_i are the parameters and λ is the regularization weight) to the training objective function, acting as a form of feature selection. L1 regularization encourages sparsity of the parameter values driving to zero the parameters that are not useful. Second, dropout is applied between the layers of both the encoder and the decoder. Dropout [9] consists in randomly omitting units (only) during training. Such a technique approximates training of many neural networks with different architectures in parallel and averaging the result. Finally, the third approach is the use of early stopping strategy in training. Basically, the performance of the model on a validation set is tracked during training; then, the training is stopped when the performance does not improve but worsens. This not only speeds up training, but also acts as a regularizer because it stops the training when it starts to overfit the data.

2.3. Case study

The proposed model was trained and tested on half-hourly load data for forecasting with a 24 h horizon. Data were obtained from a public dataset referring to an individual residential household located in Sceaux, France, with a one-minute sampling rate over a period of almost 4 years between December 2006 and November 2010 (IHEPC dataset) [8]. The dataset contains timestamped values for aggregated active and reactive power consumption and other electrical measurements, as well as some sub-metering values, as described in Table 1.

The power consumption dataset does not contain weather measurements. Therefore, the temperature measurements for the same location and time period at one-hour intervals were obtained from the Photovoltaic Geographical Information System (PVGIS) service [11].

Next subsections describe the experimental setup of the case study from data preparation to model implementation, training and tuning, and to result evaluation.

2.3.1. Data preparation

The data had to be prepared to be used for training the proposed model, as summarized in Fig. 2. The supplementary materials contain the code to download the data and run the data preparation steps reported in this subsection.

The IHEPC dataset contains some missing values in the measurements (nearly 1.25% of the samples). Although simple methods such as linear interpolation are adequate to fill in such missing values, they are likely to produce erroneous results where the length of missing segment is high [28]. In total, there are seven missing segments longer than 2 h, and the longest lasts 5 days. The missing segments were replaced with the mean value at the same day and time over the previous years in the dataset, if available; otherwise, data from the corresponding day of the previous week were used. The PVGIS dataset does not contain missing values. The datasets were re-sampled at 30-minute intervals to make them consistent with one another and with the chosen EMS application.

ANN training benefits from feature scaling or standardization in terms of training convergence and avoidance of local minima [22]. Therefore, the data were preprocessed as follows

$$x'_t = 2\frac{x_t - x_{min}}{x_{max} - x_{min}} - 1 \tag{2}$$

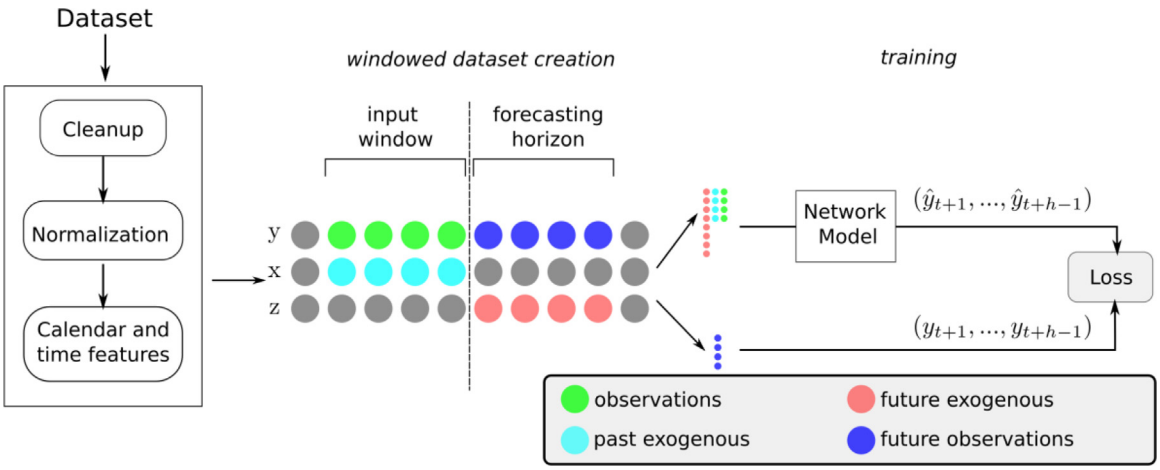

Fig. 2. Steps of experimental setup. After dataset cleanup and normalization, calendar and time features are added as exogenous features. Then, the contiguous dataset is organized into windows of data in order to train the network models by minimizing a loss function.

where x_t is the value of a variable at the generic time t, x'_t is the resulting scaled value, and x_{min} and x_{max} are the minimum and maximum values of the considered variable over the subset of the data used for training. This can be easily shown to transform a variable to be in the range $[-1, 1]$.

The time series data had then to be transformed into continuous windows of data to be used for supervised training of the model. Consecutive windows were split as (input, target) tuples, where the input consisted of the observed values of the time series and the past and future exogenous variables, as depicted in Fig. 2.

Global active power was considered as the time series to be forecasted (the output of the network). The other variables contained in the IHEPC dataset and the ambient temperature from PVGIS dataset were used as past exogenous variables. Calendar and time features were used as future exogenous variables. In particular, weekend and holiday dummy variables and Fourier terms for seasonality were used. These last terms are preferred to dummy variables for long seasonal periods as discussed in [12] where the following definition is also provided:

$$x_{sin,i,t} = \sin\left(\frac{i2\pi t}{m}\right), x_{cos,i,t} = \cos\left(\frac{i2\pi t}{m}\right), \ i \in [1, \ldots, n] \tag{3}$$

where m is the seasonal period and n is the number of terms to add. For the proposed model, daily and yearly seasonality were used with m equal to 48 and to 48×365, respectively. The number of terms n is a model hyperparameter and it was selected as described in Section 2.3.3.

2.3.2. Models used for comparison

The proposed approach was compared against three other models commonly used in the literature for short-term load demand forecasting.

The first model is the Nonlinear AutoRegressive with eXogenous inputs (NARX) ANN [6]. The NARX is trained using observations of the considered time series and future and past values of the exogenous variables. However, the same set of variables must be used for past and future values. Therefore, only calendar and time features were used as exogenous features for the NARX in the experiments.

The second model is a stack of LSTM layers that outputs a fixed-size forecasting, as in [16] (called Block LSTM model in the rest of the paper). This model receives as input observed values of the considered time series and observed (i.e., past) values of the exogenous variables; it cannot consider future exogenous variables.

Finally, the last model is a simple forecasting method used as a baseline. The model is called Naïve seasonal [12] and assumes that the considered time series has a seasonality period equal to m, in this case $m = 48$. It sets each forecast to be equal to the last observed value from the same season (i.e., the same time of day):

$$\widehat{y_i} = \mathrm{y_{t-m}} \tag{4}$$

where $\widehat{y_i}$ is the predicted value of the time series at time t.

Table 2
Model and training hyperparameters.

Hyperparameter	NARX	Block LSTM	LSTM encoder–decoder
Number of exogenous Fourier terms n		3	
Initial learning rate	1.9e−3	8.6e−3	1.9e−3
Training optimizer		Adam	
Output window size	1	48	48
Input window size	48	48 x 3	48 x 3
Dropout rate		0.1	
L1 regularization weight		0.01	
LSTM layers	–	1	2
LSTM units	–	121	[209,85]
FC layers	2	1	1
FC units	[124,89]	484	44

2.3.3. Model training and hyperparameters tuning

The proposed model and the benchmark models were trained minimizing the Mean Squared Error (MSE) loss [21]:

$$MSE = \frac{1}{N}\sum_{i=0}^{N-1}(\widehat{y_i} - y_i)^2 \tag{5}$$

where N is the number of samples, and $\widehat{y_i}$ and y_i are the predicted and observed time series values, respectively.

The dataset was split into four sequential parts. Following common best practices used in the time series forecasting literature for small datasets, the first 70% of the data was used as the training set; the next 30% was split equally into validation set, development set, and test set. This way, out-of-sample validation data were used for hyperparameter tuning. The development set was used for model architecture comparison and selection. Finally, the selected models were retrained adding the validation and development sets to the training set, and then they were tested on the test set obtaining the results reported in Section 3. It is worth pointing out that the dataset was divided before training any model in order to avoid reporting biased test results.

The considered models were implemented using Keras 2.7 [4] with Tensorflow 2.7 [1] backend. Model and training hyperparameters (i.e., variables set prior to the application of the learning algorithm, not directly selected by the algorithm itself) were tuned through a two-step procedure. At first, manual search with coordinate descent (changing only one hyper-parameter at a time, always making a change from the best configuration of hyper-parameters found up to now) was used to select a good starting point, focusing particularly on the selection of the initial learning rate. Then random search, implemented through Keras Tuner [26] was used to fine tune a subset of the hyperparameters. The resulting hyperparameters for each model and its training are reported in Table 2. The subset of hyperparameters of the proposed model tuned through random search is reported in Table 3 together with the corresponding values of the 10 best trials. All the evaluated values of the hyperparameters for the proposed model and for the models used for comparison are reported in the supplementary materials.

2.3.4. Metrics for evaluation of forecasting results

The forecasting error was measured according to several metrics described hereinafter. The Mean Absolute Error (MAE), a scale-dependent error [12], is expressed as:

$$MAE = \frac{1}{N}\sum_{i=0}^{N-1}|\widehat{y_i} - y_i| \tag{6}$$

The Normalized Root Mean Squared Error (NRMSE) and the Mean Absolute Percentage Error (MAPE) are percentage errors; as such, they can be useful to compare forecasting errors across different datasets. Their

Table 3
Ten best trials of the random search algorithm used to tune the proposed model hyperparameters. The resulting NRMSE is reported in the last column.

Initial learning rate	LSTM layers	LSTM units	FC layers	FC units	NRMSE
1.9E−3	2	[209, 85]	1	44	0.1184
3.3E−3	1	40	2	[51,424]	0.11867
7.9E−4	2	[195, 85]	1	101	0.1188
1.9E−3	2	[303, 168]	1	277	0.1188
8.7E−4	2	[408, 87]	1	57	0.1188
5.4E−4	3	[139, 497, 178]	2	[484, 284]	0.1190
7.3E−4	2	[60, 46]	2	[225, 190]	0.1191
3.7E−4	2	[419, 64]	2	[496, 148]	0.1192
1.7E−3	1	141	1	243	0.1194
5.9E−4	3	[35, 271, 381]	2	[34, 58]	0.1194

expression can be found in [21] and is as follows:

$$NRMSE = \frac{\sqrt{MSE}}{\max_i y_i - \min_i y_i} 100 \tag{7}$$

$$MAPE = \frac{1}{N} \sum_{i=0}^{N-1} \left| \frac{y_i - \widehat{y_i}}{y_i} \right| 100 \tag{8}$$

Finally, it is useful to consider scaled metrics [13] that scale the error by the training MAE of simple forecasting methods. The Mean Absolute Scaled Error (MASE), considering Naïve Seasonal forecasting as the base method, is defined in [13] and expressed as:

$$MASE = \frac{1}{N} \frac{\sum_{i=0}^{N-1} |\widehat{y_i} - y_i|}{\frac{1}{T-m} \sum_{t=m}^{T} |y_t - y_{t-m}|} \tag{9}$$

where T is the length of the training set and m is the seasonality period.

For short-term forecasting based on ANN models it is uncommon to retrain the model every time a forecasting is produced. Especially when intended for use by EMSs, a model is trained once and used repeatedly for forecasting with a specified horizon. Therefore, to validate the model and estimate its performance on forecasting future data, backtesting without retraining is used. The procedure consists of forecasting for a fixed horizon given an input window from the test dataset and then repeating over a sliding window (see Fig. 3).

Through backtesting, a series of forecastings is obtained as it would have been at the historical time of forecasting start times. Error metrics can be calculated for each of these forecastings. Therefore, a profile of errors (daily errors for example) can be analyzed. However, it is useful to also have a single index over the entire test set (one per used metric) to characterize the overall error. Two ways are possible: statistics (like mean and variance) can be calculated over the vector of metric values; otherwise, the metrics can be calculated on the concatenation of the forecasted windows. However, this last method is sound only if the stride between backtesting windows equals the forecasting horizon. It is worth noting that, for metrics like MAE or MAPE, which are averages of point errors, the mean over the vector of metric values is equivalent to the second method, but this is not the case for NRMSE.

Often, papers in the literature just test the proposed methods on a single forecasting window. Also, when backtesting is used, and multiple forecastings are produced, it is not explained how the reported metrics are calculated. This makes the results more difficult to interpret and to reproduce. At the best of the authors' knowledge, the first method is usually adopted by practitioners, even if not explicitly stated. However, it is worth considering the alternative method as well.

Backtesting results are reported in the next section using the mean of daily metrics, and the NRMSE error index over the aggregated profile is also reported.

3. Results

Training and testing were executed on a machine with Intel Core i7-8700 CPU, 16 GB of RAM, and an Nvidia GeForce GTX 1070 GPU. Execution times for training, backtesting over the entire test set, and one-day forecasting

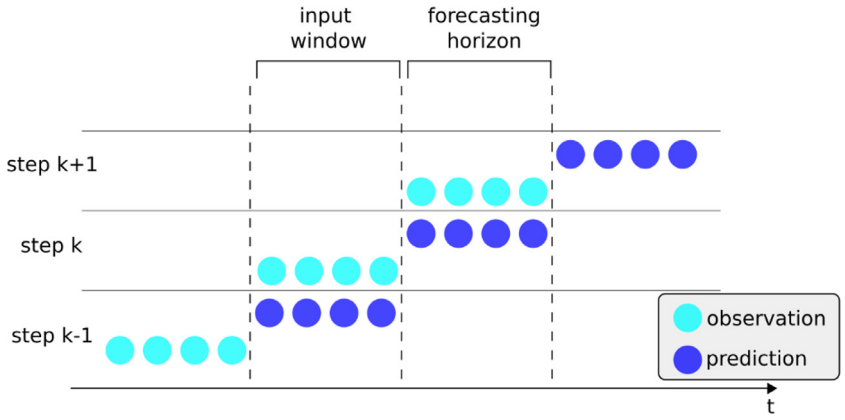

Fig. 3. Backtesting procedure. The forecasting is repeated giving as input to the model the observations in the input window and obtaining the prediction for the forecasting horizon. Then, the input window is moved by a fixed stride (equal to the forecasting horizon in this case), and the new forecasting is produced. The model only considers observations known at the time of forecasting (For ease of representation, exogenous variables are not depicted).

Table 4
Training, backtesting, and forecasting times (averages and std deviation).

Model	Training	Backtesting	Forecasting
NARX	57.9 s ± 23 s	4.71 s ± 26 ms	157 ms ± 2 ms
Block LSTM	1 min 24 s ± 26 s	0.876 s ± 3 ms	146 ms ± 4 ms
Prop. model	2 min 6 s ± 13 s	1.71 s ± 8 ms	165 ms ± 1 ms

are reported in Table 4 and are calculated over 10 executions. It can be observed that the one-day forecasting time of the proposed model is compatible with the time-steps of execution of forecasting-based EMSs, which are in the order of minutes. The proposed model can be exported after training and be deployed on a forecasting model running on a low-cost single board computer as in [19]. Also, the used third-party libraries are released under open source licenses. Therefore, the deployment on a working environment would not require high costs. However, this aspect is outside the scope of the present work.

The mean values of the daily metrics are reported in Fig. 4. It can be observed that the proposed model outperforms the other models according to all the metrics. In particular, there is a reduction of about 8% on MASE with respect to NARX and Block LSTM. These results also suggest the advantage of considering both past and future exogenous variables. In fact, the NARX model can only consider variables that will be also known in the future and it cannot consider historical observations at the time of forecasting (unless an external forecasting provides the corresponding values for the forecasting horizon). The Block LSTM model, on the other hand, can consider historical observations, but cannot consider future exogenous variables. These gaps penalize the performance of the benchmark models compared with that obtained with the proposed approach.

It is also interesting to look at the profile of the daily metrics. Fig. 5 shows the daily NRMSE; it can be observed that the proposed model exhibits a smaller error almost every day. Furthermore, it is possible to observe that the daily NRMSE of the proposed model does not show an increasing trend that would have implied the need for retraining the model. Therefore, the forecasting model can be used by an EMS without retraining for month-long periods.

The NRMSE calculated over the aggregated forecasting results is reported in Table 5. Even in this case, the proposed model outperforms the other benchmark models. It can be observed that the NRMSE calculated over the aggregated result is smaller than the mean of daily NRMSE; this result is due to the calculation of the metric over a longer period which makes it more tolerant to punctual errors.

As an example, the forecasting result of three consecutive days is plotted in Fig. 6. It can be observed that the load demand variability and, thus, the forecasting error are higher during the hours of higher consumption. Further work could account for this by using a probabilistic forecasting approach. Furthermore, it is worth noting

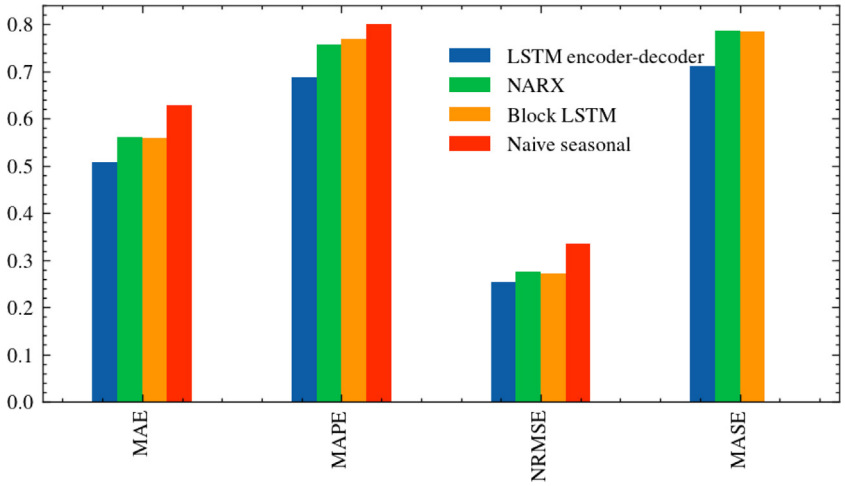

Fig. 4. Mean values of daily error metrics for each model (MASE is not calculated for Naïve seasonal because the latter is the base method used in the metric definition).

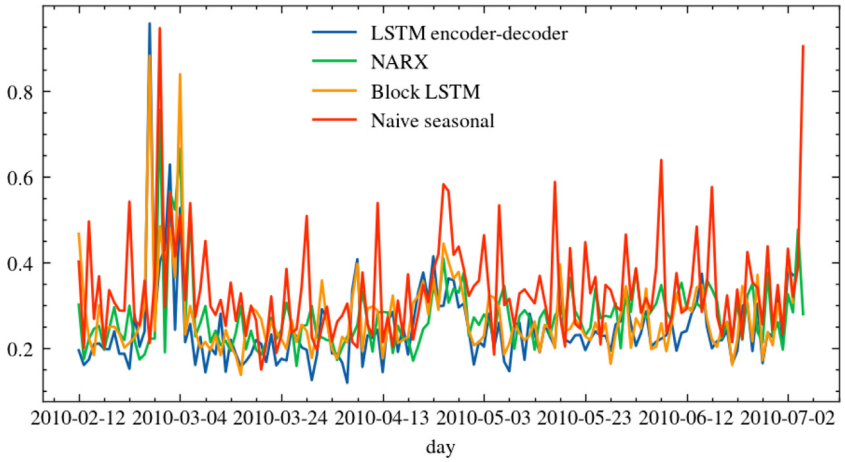

Fig. 5. Daily NRMSE of the proposed model and the other models used for comparison.

Table 5
NRMSE over aggregated forecasting results.

Model	NRMSE
Prop. model	0.1161
NARX	0.1238
Block LSTM	0.1240
Naïve seasonal	0.1521

that it is not easy to determine a cause for such errors during the hours of higher consumption. A possible future extension of the model could explore how to make it interpretable (i.e., it cannot provide explanations for its output in understandable terms to a human). For example, an attention layer [37] could be added to identify salient elements of the input sequence and therefore a possible explanation of the produced output.

The reported results show that the forecasting of the individual residential load demand is more difficult than aggregated load demand forecasting, thus justifying the application of a more complex and expressive model like the proposed one. In fact, results reported in the literature for aggregated load demand reach lower error metrics

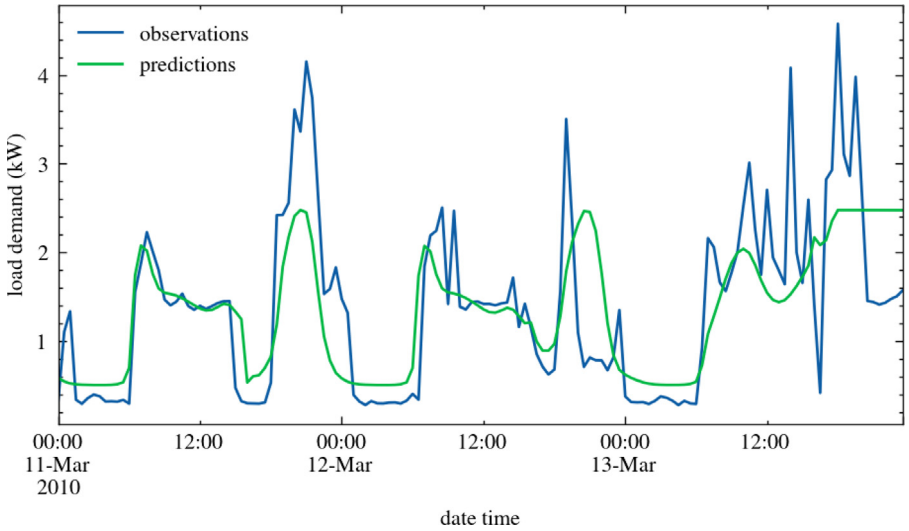

Fig. 6. Plot of three sample days of forecasting results obtained using the proposed model.

using simpler methods for example in [6]. Furthermore, the proposed deep learning approach produces good results without the need for hand-coded feature extraction or specific domain knowledge, making the approach easily applicable to other datasets. In addition, the flexibility of the model allows the inclusion of all the available variables in the dataset. Finally, it is worth pointing out that the proposed model is tailored for EMS applications, hence these results reflect the forecasting errors that an EMS would experience in its practical application and would need to mitigate. Considering that EMSs proposed in the literature (e.g., in [18]) are able to mitigate errors equal to or higher than those reported in this paper, the proposed forecasting method proved to be fully effective for use within an EMS.

Finally, it should be noted that all the tests were conducted using fixed-length forecasting horizon. However, the proposed model can produce variable-length forecasting, but its evaluation and validation using backtesting deserves further study, in particular with regard to the error aggregation procedure because it cannot be simply applied as in this paper.

4. Conclusions

The paper presented a model for day-ahead residential load demand forecasting that is suitable for energy management systems. The proposed model considers load demand observations and both past and future exogenous variables through a stack of LSTM cells that acts as an encoder and a stack of LSTM cells that acts as a decoder. The output of the decoder is transformed through a fully connected network into a forecasting for the following day. The proposed model was designed considering individual household load demand time series, which have higher variability with respect to utility-level load demand time series; furthermore, it was tuned, trained, and tested on a publicly available dataset. However, no other assumptions were made, and the model can be readily tuned and trained on other individual household load demand datasets.

With respect to the other considered models, the proposed one has fewer limitations. In fact, it can consider different sets of variables as past and future exogenous variables, and it also can produce variable length forecasting.

To be used by EMSs, day-ahead forecasting is produced regularly without usually retraining the model. Therefore, the proposed approach was evaluated through backtesting without retraining. The proposed model outperformed three selected benchmark methods on all the considered error metrics. Furthermore, a discussion about the appropriate way of reporting scalar values for the error metrics on a backtesting result was presented, and two alternative approaches were proposed.

Finally, it is worth observing that EMSs benefit from good forecasting, but they still need to account for and mitigate forecasting errors. Therefore, their optimization formulation could benefit from probabilistic forecasting with prediction intervals. Future work will extend the proposed model to also provide prediction intervals.

Funding

This research was funded by Italian Ministry of University and Research (MUR), Italy under the following grants: grant number 2017WA5ZT3_004, program PRIN2017, project HEROGRIDS; Sustainable Mobility Center (CNMS) Spoke 3, project id CN00000023, CUP: B43C22000440001.

Appendix A. Supplementary materials

Data preparation. Folder containing a Jupyter notebook and Python code to download the data and execute the data preparation steps described in Section 2.3.1.

Hyperparameters-tuning-results. Folder containing the full results of the random search algorithm used for tuning the hyperparameters of the proposed models and of the NARX and Block LSTM models used for comparison in the paper.

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.matcom.2023.06.017.

References

[1] M. Abadi, et al., TensorFlow: large-scale machine learning on heterogeneous systems, 2015, [Online]. Available: https://www.tensorflow.org/.

[2] M. Alhussein, K. Aurangzeb, S.I. Haider, Hybrid CNN-LSTM model for short-term individual household load forecasting, IEEE Access 8 (2020) 180544–180557, http://dx.doi.org/10.1109/ACCESS.2020.3028281.

[3] K. Amarasinghe, D.L. Marino, M. Manic, Deep neural networks for energy load forecasting, in: 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), 2017, pp. 1483–1488, http://dx.doi.org/10.1109/ISIE.2017.8001465.

[4] F. Chollet, et al., Keras, 2015.

[5] G. Di Lorenzo, L. Martirano, R. Araneo, G. Petrone, Modeling and design of a residential energy community with PV sharing, in: Proceedings - 2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe, EEEIC/ I and CPS Europe 2020, 2020, http://dx.doi.org/10.1109/EEEIC/ICPSEurope49358.2020.9160650.

[6] A. Di Piazza, M.C. Di Piazza, G. La Tona, M. Luna, An artificial neural network-based forecasting model of energy-related time series for electrical grid management, Math. Comput. Simulation 184 (2021) 294–305, http://dx.doi.org/10.1016/j.matcom.2020.05.010.

[7] K. Greff, R.K. Srivastava, J. Koutnik, B.R. Steunebrink, J. Schmidhuber, LSTM: A search space odyssey, IEEE Trans. Neural Netw. Learn. Syst. 28 (10) (2017) 2222–2232, http://dx.doi.org/10.1109/TNNLS.2016.2582924.

[8] G. Hebrail, A. Berards, Individual household electric power consumption data set. UCI machine learning repository, 2012, https://archive.ics.uci.edu/ml/datasets/individual+household+electric+power+consumption, (Accessed 29 January 2022).

[9] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R.R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, 2012, pp. 1–18, [Online]. Available: http://arxiv.org/abs/1207.0580.

[10] T. Hong, S. Fan, Probabilistic electric load forecasting: A tutorial review, Int. J. Forecast. 32 (3) (2016) 914–938, http://dx.doi.org/10.1016/j.ijforecast.2015.11.011.

[11] T. Huld, R. Müller, A. Gambardella, A new solar radiation database for estimating PV performance in Europe and Africa, Sol. Energy 86 (6) (2012) 1803–1815, http://dx.doi.org/10.1016/j.solener.2012.03.006.

[12] R.J. Hyndman, G. Athanasopoulos, Forecasting: principles and practice, 3rd editio, OTexts, Melbourn, Australia, 2021, [Online]. Available: OTexts.com/fpp3.

[13] R.J. Hyndman, A.B. Koehler, Another look at measures of forecast accuracy, Int. J. Forecast. 22 (4) (2006) 679–688, http://dx.doi.org/10.1016/j.ijforecast.2006.03.001.

[14] N. Khan, I.U. Haq, S.U. Khan, S. Rho, M.Y. Lee, S.W. Baik, DB-Net: A novel dilated CNN based multi-step forecasting model for power consumption in integrated local energy systems, Int. J. Electr. Power Energy Syst. 133 (2021) 107023, http://dx.doi.org/10.1016/j.ijepes.2021.107023.

[15] T.-Y. Kim, S.-B. Cho, Predicting residential energy consumption using CNN-LSTM neural networks, Energy 182 (2019) 72–81, http://dx.doi.org/10.1016/j.energy.2019.05.230.

[16] W. Kong, Z.Y. Dong, Y. Jia, D.J. Hill, Y. Xu, Y. Zhang, Short-term residential load forecasting based on LSTM recurrent neural network, IEEE Trans. Smart Grid 10 (1) (2019) 841–851, http://dx.doi.org/10.1109/TSG.2017.2753802.

[17] S. Kumar, L. Hussain, S. Banarjee, M. Reza, Energy Load Forecasting using Deep Learning Approach-LSTM and GRU in Spark Cluster, in: 2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT), 2018, pp. 1–4, http://dx.doi.org/10.1109/EAIT.2018.8470406.

[18] G. La Tona, M.C. Di Piazza, M. Luna, Effect of daily forecasting frequency on rolling-horizon-based EMS reducing electrical demand uncertainty in microgrids, Energies 14 (6) (2021) 1598, http://dx.doi.org/10.3390/en14061598.

[19] G. La Tona, M. Luna, A. Di Piazza, M.C. Di Piazza, Development of a forecasting module based on tensorflow for use in energy management systems, in: IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society, 2019, pp. 3063–3068, http://dx.doi.org/10.1109/IECON.2019.8926801.

[20] G. La Tona, M. Luna, A. Di Piazza, M.C. Di Piazza, Towards the real-world deployment of a smart home EMS: A DP implementation on the raspberry Pi, Appl. Sci. 9 (10) (2019) 2120, http://dx.doi.org/10.3390/app9102120.

[21] T. Le, M.T. Vo, B. Vo, E. Hwang, S. Rho, S.W. Baik, Improving electric energy consumption prediction using CNN and bi-LSTM, Appl. Sci. 9 (20) (2019) 4237, http://dx.doi.org/10.3390/app9204237.
[22] Y.A. LeCun, L. Bottou, G.B. Orr, K.-R. Müller, Efficient. BackProp, Efficient backprop, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, in: LECTU, vol. 7700, (1998) 2012, pp. 9–48, http://dx.doi.org/10.1007/978-3-642-35289-8_3.
[23] S. Makridakis, E. Spiliotis, V. Assimakopoulos, The.M4. Competition:, 100000 Time series and 61 forecasting methods, Int. J. Forecasting 36 (1) (2020) 54–74, http://dx.doi.org/10.1016/j.ijforecast.2019.04.014.
[24] D.L. Marino, K. Amarasinghe, M. Manic, Building energy load forecasting using Deep Neural Networks, in: IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, 2016, pp. 7046–7051, http://dx.doi.org/10.1109/IECON.2016.7793413.
[25] E. Mocanu, P.H. Nguyen, M. Gibescu, W.L. Kling, Deep learning for estimating building energy consumption, Sustain. Energy Grids Netw. 6 (2016) 91–99, http://dx.doi.org/10.1016/j.segan.2016.02.005.
[26] T. O'Malley, et al., "Kerastuner.", 2019.
[27] B.N. Oreshkin, D. Carpov, N. Chapados, Y. Bengio, N-BEATS: Neural basis expansion analysis for interpretable time series forecasting, 2019, pp. 1–31, [Online]. Available: http://arxiv.org/abs/1905.10437.
[28] A. Rahman, V. Srikumar, A.D. Smith, Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks, Appl. Energy 212 (2018) 372–385, http://dx.doi.org/10.1016/J.APENERGY.2017.12.051.
[29] R. Rajabi, A. Estebsari, Deep learning based forecasting of individual residential loads using recurrence plots, in: 2019 IEEE Milan PowerTech PowerTech 2019, 2019, http://dx.doi.org/10.1109/PTC.2019.8810899.
[30] M. Sajjad others, A novel CNN-GRU-based hybrid approach for short-term residential load forecasting, IEEE Access 8 (2020) 143759–143768, http://dx.doi.org/10.1109/ACCESS.2020.3009537.
[31] L. Sehovac, C. Nesen, K. Grolinger, Forecasting Building Energy Consumption with Deep Learning: A Sequence to Sequence Approach, in: 2019 IEEE International Congress on Internet of Things (ICIOT), 2019, pp. 108–116, http://dx.doi.org/10.1109/ICIOT.2019.00029.
[32] H. Shi, M. Xu, R. Li, Deep learning for household load forecasting—A novel pooling deep RNN, IEEE Trans. Smart Grid 9 (5) (2018) 5271–5280, http://dx.doi.org/10.1109/TSG.2017.2686012.
[33] H. Shi, M. Xu, Q. Ma, C. Zhang, R. Li, F. Li, A whole system assessment of novel deep learning approach on short-term load forecasting, Energy Procedia 142 (2017) 2791–2796, http://dx.doi.org/10.1016/j.egypro.2017.12.423.
[34] M. Suresh, M.S. Anbarasi, J. Divyabharathi, D. Harshavardeni, S. Meena, Household electricity power consumption prediction using CNN-GRU techniques, in: 2021 International Conference on System Computation, Automation and Networking, ICSCAN, 2021, http://dx.doi.org/10.1109/ICSCAN53069.2021.9526485.
[35] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, Adv. Neural Inform. Process. Syst. 4 (January) (2014) 3104–3112, [Online]. Available: http://arxiv.org/abs/1409.3215.
[36] F.U.M. Ullah, A. Ullah, I.U. Haq, S. Rho, S.W. Baik, Short-term prediction of residential power energy consumption via CNN and multi-layer bi-directional LSTM networks, IEEE Access 8 (2020) 123369–123380, http://dx.doi.org/10.1109/ACCESS.2019.2963045.
[37] A. Vaswani, et al., Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.
[38] H. Wilms, M. Cupelli, A. Monti, Combining auto-regression with exogenous variables in sequence-to-sequence recurrent neural networks for short-term load forecasting, in: 2018 IEEE 16th International Conference on Industrial Informatics (INDIN), 2018, pp. 673–679, http://dx.doi.org/10.1109/INDIN.2018.8471953.