Weather Based Day-Ahead and Week-Ahead Load Forecasting using Deep Recurrent Neural Network

Mingzhe Zou, Duo Fang, Gareth Harrison, Sasa Djokic
School of Engineering
The University of Edinburgh
Edinburgh, UK
sasa.djokic@ed.ac.uk

***Abstract*—The main purpose of load forecasting is to provide an accurate estimation of the future electricity demands, which is becoming increasingly important for the operation and planning of existing electricity network and future smart grids, as they will feature much higher ranges of uncertainties, larger variations of power flows and increased levels of interactions between supply and demand sides. Typical load profiles exhibit periodicity, allowing to extract patterns from demand time series and available historical recordings. However, there are many factors that cause strong variations of these demand patterns, including calendar and socio-behavioral aspects (time of the day, day of the week, season of the year, but also weekly working schedule, public holidays, etc.), as well as meteorological or weather related factors (ambient temperature, solar irradiation, precipitation, etc.). This paper analyzes load forecasting using a stacked bidirectional long short-term memory (SB-LSTM) recurrent neural network based approach, which is a state-of-the-art method for regression analysis of time-series data under deep learning framework. The analysis is performed on a case study of residential demands in Scotland, for which a five-year-length datasets containing both load and weather data recordings are available. The presented results and analysis allow to evaluate how accurately SB-LSTM approach can provide predictions for both day-ahead and week-ahead load forecasting, taking into account meteorological information.**

Keywords—Day-ahead and week-ahead load forecasting, deep learning, meteorological data, recurrent neural network, stacked bidirectional long short-term memory approach.

I. INTRODUCTION

Load forecasting is generally aimed at providing an accurate estimation of the future electricity demands, based on available historical recordings and other relevant data. Load forecasting is becoming increasingly important for the operation and planning of existing electricity network and future smart grids, which will feature much higher ranges of uncertainties, larger variations of power flows and increased levels of interactions between supply and demand sides. In this context, long-term forecasting is important for power system planning, while short-term forecasting, especially day-ahead forecasting, is essential for operational studies [1].

There are many load forecasting approaches in existing literature, including time series analysis [2], Gaussian process [3], neural network [1], [4], fuzzy regression [5], tree-based model [6], support vector regression [7], etc. Typical load profiles exhibit periodicity, but also feature strong hourly, daily, weekly and monthly/seasonal variations. Weather (meteorological) factors, such as temperature, precipitation, solar irradiation, as well as socio-behavioral factors, all have impact on demand variations. Inclusion of these factors will improve forecasting accuracy, but will increase dimensions of forecasting problem, as synchronous series of calendar and weather variables should be analyzed together with demand series. Additional issue is that many current approaches lack flexibilities in terms of forecasting over different time periods, e.g. a day-ahead forecasting vs week-ahead forecasting.

In recent years, deep learning models are playing an increasingly important role in many areas. Compared to traditional machine learning models (e.g., classic artificial neural network with only one hidden layer), deep learning models usually have a cascade of multiple layers, through which the analyzed data are transformed, allowing to extract their features more efficiently. In relatively simple problems, or if there are not enough data, deep learning may not even be as accurate as classic machine learning, but its learning ability is much stronger in complex problems with huge amounts of available data. Long short-term memory (LSTM) is a type of a recurrent neural network (RNN) belonging to deep learning framework, which has been proven to be well-suited for classification and regression of time series data since it was first introduced in [8]. LSTM is also a very flexible model, as its inputs and outputs can be either a single value, or a sequence, and it is, therefore, suitable for "one-to-one", "one-to-many", "many-to-one" and "many-to-many" problems.

This paper analyzes performance of a stacked bidirectional LSTM (SB-LSTM) approach for both day-ahead and week-ahead load forecasting. Section II presents traditional recurrent neural network model and compares it with LSTM. Section III generalizes SB-LSTM models for day-ahead and week-ahead load forecasting and provides comparison of the results, while Section IV gives main conclusions.

II. LSTM RECURRENT NEURAL NETWORK

In statistical modeling, regression analysis is used to estimate relationships between "inputs" and "outputs", defined as predictor and response variables. Regression analysis can be used for forecasting, if variables of past and current states are set as predictors, while variables in the future states are set as responses. Neural network-based methods are widely used in classification problems as supervised learning approaches and they can also be implemented for regression analysis, by setting their outputs as response variables instead as categories.

978-1-7281-3815-2/19/$31.00 ©2019 IEEE

Authorized licensed use limited to: Indian Institute of Technology (Ropar). Downloaded on May 23,2020 at 12:20:19 UTC from IEEE Xplore. Restrictions apply.

A. Recurrent Neural Network (RNN)

Recurrent neural network (RNN) is a class of neuron network model specialized for sequence data, which is widely used in natural language processing (NLP), speech recognition, machine translation and other applications. Unlike feedforward neural networks, RNN is designed to memorize sequence data. Fig. 1 shows a typical architecture of a traditional RNN. It can be seen that each RNN cell uses both the previous internal state h_{t-1} and its current input x_t to predict y_t, where internal state h_t is updated and passed to the next step.

When handling long data time series, traditional RNN tends to have "gradient vanishing" and "gradient exploding" issues [9]. The training process of traditional RNN uses the Backpropagation Through Time (BPTT) algorithms to calculate the gradients and propagation of error, which need to be multiplied with weights in all steps [10]. In learning long-length dependencies, if the weights are smaller than 1, then the error will be very close to zero after the backpropagation (i.e., gradient vanishing), and if the weights are larger than 1, then the error will be too large at the end (i.e., gradient exploding).

B. Long Short-Term Memory (LSTM) Cell

To deal with the mentioned gradient-based problems, long short-term memory (LSTM) method is developed, which is also relatively insensitive to gaps of duration between important sequence events [8]. Every cell of LSTM has three gates to control whether to remember or to forget certain information and whether to output it at a specific point. The architecture of the LSTM cell is illustrated in Fig. 2.

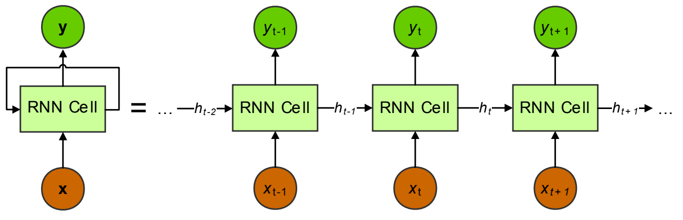

Fig. 1. Typical traditional RNN architecture

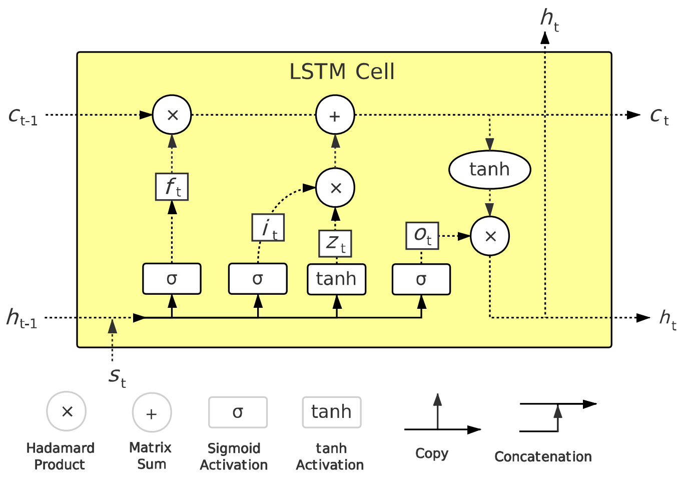

Fig. 2. LSTM cell architecture (solid arrows represent weighed vectors, dot arrows represent unweighted vectors)

In Fig. 2, s_t is the current input and h_t is its output. Like traditional RNN, LSTM can also pass the internal states, but an LSTM cell has two different states. State c is the cell state that changes extremely slow and in most cases it is very similar to the previous cell state. State h stands for the hidden state, which varies a lot from nodes to nodes and usually depends on the current input, as well as on the previous hidden state. Three gate signals are generated by applying activation function to weighted matrix, concatenating the current input and the previous hidden state:

Forget Gate: Decides which part of the cell state information should be forgotten. This gate uses h_{t-1} and s_t as inputs and it is activated by sigmoid activation function σ. It can be formulated as:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, s_t] + b_f\right) \tag{1}$$

Remember Gate: Controls which parts of the input and hidden states should be remembered. This gate signal i is activated by sigmoid activation function and another candidate vector z is activated by *tanh* function. The formulations are:

$$\begin{cases} i_t = \sigma\left(W_i \cdot [h_{t-1}, s_t] + b_i\right) \\ z_t = \tanh\left(W_z \cdot [h_{t-1}, s_t] + b_z\right) \\ c_t = f_t \times c_{t-1} + i_t \times z_t \end{cases} \tag{2}$$

Output Gate: Decides which part of the information to output, where the output information is again activated by *tanh* activation function, which can be formulated as:

$$\begin{cases} o_t = \sigma\left(W_o \cdot [h_{t-1}, s_t] + b_o\right) \\ h_t = o_t \times \tanh\left(c_t\right) \end{cases} \tag{3}$$

In (1)-(3), W_f, W_i, W_z, W_o, and b_f, b_i, b_z and b_o are weight and bias parameters of each gate. These three gates enable LSTM to remember important information and forget the irrelevant one over a long sequence of data. However, this significantly increases the magnitudes of the hyperparameters, which makes LSTM hard to tune and train.

C. Stacked Bidirectional Long Short-term Memory (SB-LSTM)

The original LSTM architecture has only one hidden layer of one direction, which may not perform well for deep information mining. The stacked bidirectional LSTM (SB-LSTM) has multiple hidden bidirectional LSTM layers, where each bidirectional LSTM layer contains two hidden layers of opposite directions to the same outputs. The schematic of a bidirectional long short-term memory layer is plotted in Fig. 3. The bidirectional structure is firstly introduced in [11], with the main idea to make the outputs to have both forward and backward information. The stacked structure allows for a deeper network model, which, therefore, could learn more efficiently and accurately in case of more complex and more challenging problems [12].

Authorized licensed use limited to: Indian Institute of Technology (Ropar). Downloaded on May 23,2020 at 12:20:19 UTC from IEEE Xplore. Restrictions apply.

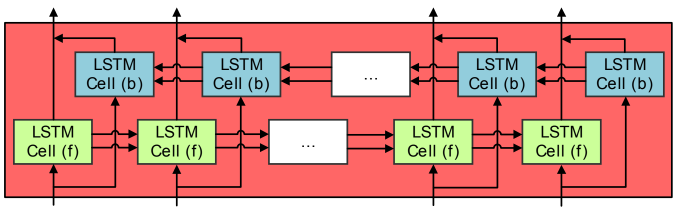

Fig. 3. A bidirectional long short-term memory layer

In case of the extremely complex problems, the depth of the SB-LSTM is usually increased by adding bidirectional LSTM layers. However, a deeper SB-LSTM is more likely to have overfitting issue. To resolve that, many methods are proposed, for example, a "dropout" has been proven to be an efficient regularization technique, which refers to random dropping out of hidden and visible neurons in a certain layer [13]. The L2 regularization can also improve the accuracy of the model by adding the squared values of weight coefficients to the loss function as penalty factors, which then makes the network independent of certain weights parameters [14]. The formulation of L2 regularization can be written as:

$$\varepsilon = f\left(y,\hat{y}\right)+\lambda\sum_{i=1}^{k}\left|w_i\right|^2 \tag{4}$$

where ε is the final total error, $f(\ )$ is the error function, y and $\hat{y}$ are the actual value and model results, respectively, w_i is the i-th weight parameter, and λ is an additional factor.

III. Case Study

This section describes the generalizations of the SB-LSTM models used for both day-ahead and week-ahead load forecasting of a residential demand in a town in Scotland, for which five-year demand and weather data recordings are available [15], [16]. Demand measurements are average 30-minute values of active power (**AP**), while meteorological data (**MD**) are: average 60-minute recordings of temperature (**T**), solar irradiation (**S**), precipitation (**P**) and wind speed (**WS**). Since the temporal resolutions of load and meteorological data are different, the meteorological series are all up-sampled to 48 values per day using the spline interpolation method, in order to make them synchronous with the demand data [17].

A. Calendar Variables

Demand recordings have strong autocorrelations and temporal dependencies, where actual variations of considered electrical load are influenced by standard calendar variables (**SCV**), i.e., year, month, day, hour and day of the week (**YY**, **MM**, **DD**, **HH**, **DW**), and also by public holidays (**HLD**) and British summer time (**BST**) [18], [19]. Typical daily load profiles are illustrated in Fig. 4, showing calculated average daily load profiles of the same day-types. In addition, solar analemma (Sun azimuth angle, **SA**, and Sun elevation angle, **SE**) can be used to precisely represent time of the day and season of the year variations and are calculated as in [20]. Finally, Moon's rotation also has impacts on Earth and human behavior [21], so moon phase (**MP**) is taken into consideration as the last additional variable [22].

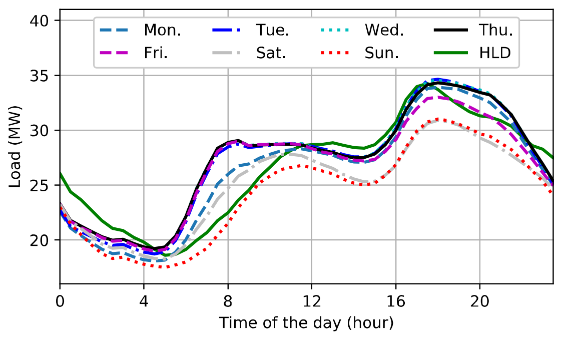

Fig. 4. Average daily load profiles of different types of days

B. Data Preprocessing: Dataset Splitting, One Hot Encoder, and Data Normalization

The main purpose of this paper is to compare day-ahead and week-ahead load forecasting (i.e., 48 load points for the day-ahead forecasting and 48×7=336 load points for the week-ahead forecasting) using historical load data and day/week-ahead meteorological forecast information as inputs. The whole dataset is divided into training set, validation set and test set, where the minimum demand week, the average demand week and the maximum demand week in the fifth year are selected for testing, while contiguous 4-year-length data exactly before the test set are used as training/validation set, as Fig. 5 shows. The training set and validation sets are then distinguished by splitting the 4-year-length data by the ratio of 0.8 : 0.2 in contiguous blocks for every 100 days. That is, in training/validation set, day $1+100\times i$ to day $80+100\times i$ are used for training, and day $81+100\times i$ to day $100\times(i+1)$ are validation set, where i is a non-negative integer. The dataset should not be divided randomly, as the recordings are temporally dependent.

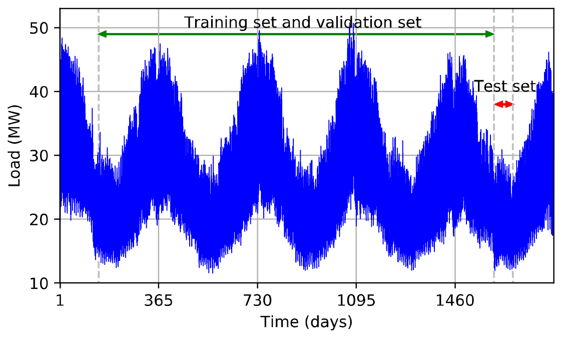

Fig. 5. Illustration of dataset splitting into training set, validation set and test set

Standard calendar variables (**SCV**) are inherently categorical features and, therefore, overall accuracy of neural network based models can be improved by firstly encoding categorical variables using one hot encoder [23]. Analemma, Moon phase, meteorological and load recordings are scaled and mapped into the range [-1, 1] using min-max normalizations:

$$\mathbf{DS}_{\text{new}} = \min\left(\mathbf{DS}_{\text{new}}\right)+\frac{\max\left(\mathbf{DS}_{\text{new}}\right)-\min\left(\mathbf{DS}_{\text{new}}\right)}{\max\left(\mathbf{DS}_{\text{old}}\right)-\min\left(\mathbf{DS}_{\text{old}}\right)}\times \left(\mathbf{DS}_{\text{old}}-\min\left(\mathbf{DS}_{\text{old}}\right)\right) \tag{5}$$

Authorized licensed use limited to: Indian Institute of Technology (Ropar). Downloaded on May 23,2020 at 12:20:19 UTC from IEEE Xplore. Restrictions apply.

where $\mathbf{DS}_{\mathrm{old}}$ and $\mathbf{DS}_{\mathrm{new}}$ are the original and transformed data series, respectively. The minimum and maximum values of the original data series are assumed to be the minimum and maximum recordings in the training/validation set, which does not consider any information from the test set. Z-score standardization method is not used, because the distributions of variables are quite different, and some variables may have relatively large magnitudes after Z-score standardization, affecting the training process.

C. Generalization of SB-LSTM Model

One advantage of SB-LSTM method compared to classical artificial feedforward neural network is that it can analyze the whole input sequence natively and learn its temporal dependencies. Therefore, the characteristics of the load series need not be artificially extracted, which may cause loss of some important hidden information. As the output of SB-LSTM can be sequenced data, the presented SB-LSTM method is designed for analyzing "many-to-many" regression problem.

In the day-ahead and week-ahead forecasting problems, the SB-LSTM models use the following sequence data as inputs:

1) Calendar variables (including standard calendar variables, analemma (Sun azimuth and elevation angles) and Moon phase) of the forecasted day/week, i.e., ($\mathbf{SCV}_{+1\ or\ +1\sim7}$, $\mathbf{SA}_{+1\ or\ +1\sim7}$, $\mathbf{SE}_{+1\ or\ +1\sim7}$, $\mathbf{MP}_{+1\ or\ +1\sim7}$), where $\mathbf{SCV}_{+1\ or\ +1\sim7}$ = ($\mathbf{YY}_{+1\ or\ +1\sim7}$, $\mathbf{MM}_{+1\ or\ +1\sim7}$, $\mathbf{DD}_{+1\ or\ +1\sim7}$, $\mathbf{HH}_{+1\ or\ +1\sim7}$, $\mathbf{DW}_{+1\ or\ +1\sim7}$, $\mathbf{HLD}_{+1\ or\ +1\sim7}$, $\mathbf{BST}_{+1\ or\ +1\sim7}$,).

2) Meteorological data (temperature, solar irradiation, precipitation and wind speed) of the forecasted day/week, i.e., $\mathbf{MD}_{+1\ or\ +1\sim7}$ = ($\mathbf{T}_{+1\ or\ +1\sim7}$, $\mathbf{S}_{+1\ or\ +1\sim7}$, $\mathbf{P}_{+1\ or\ +1\sim7}$, $\mathbf{WS}_{+1\ or\ +1\sim7}$).

3) Load recordings from past 28 days before the forecasted day/week, i.e., $\mathbf{AP}_{-27\sim0}$ = ($\mathbf{AP}_{-27}$, $\mathbf{AP}_{-26}$, $\mathbf{AP}_{-25}$, …, $\mathbf{AP}_{0}$).

The outputs of the day-ahead forecasting and week-ahead forecasting of SB-LSTM models are $\mathbf{AP}_{+1}$ and $\mathbf{AP}_{+1\sim7}$ = ($\mathbf{AP}_{+1}$, $\mathbf{AP}_{+2}$, $\mathbf{AP}_{+3}$, $\mathbf{AP}_{+4}$, $\mathbf{AP}_{+5}$, $\mathbf{AP}_{+6}$, $\mathbf{AP}_{+7}$), respectively.

Each SB-LSTM model is implemented with three stacked bidirectional LSTM layers (512 hidden units are used for every layer) and three dropout layers (dropout rate = 0.5), while L2 regularization with factor set as 0.0005 is adopted to prevent overfitting. Adam optimizer is used to minimize mean absolute error (MAE) between model results and actual recordings [24], [25]. This SB-LSTM architecture is illustrated in Fig. 6.

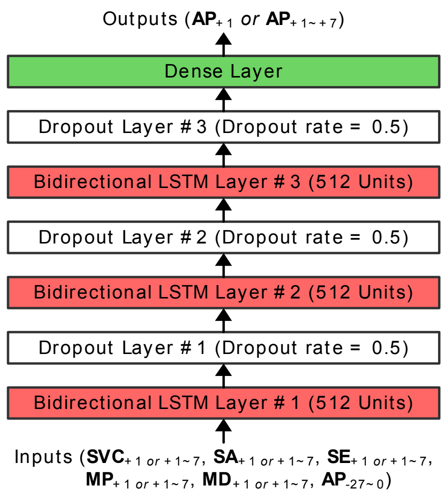
Outputs ($\mathbf{AP}_{+1}$ or $\mathbf{AP}_{+1\sim+7}$)
Dense Layer
Dropout Layer # 3 (Dropout rate = 0.5)
Bidirectional LSTM Layer # 3 (512 Units)
Dropout Layer # 2 (Dropout rate = 0.5)
Bidirectional LSTM Layer # 2 (512 Units)
Dropout Layer # 1 (Dropout rate = 0.5)
Bidirectional LSTM Layer # 1 (512 Units)
Inputs ($\mathbf{SVC}_{+1\ or\ +1\sim7}$, $\mathbf{SA}_{+1\ or\ +1\sim7}$, $\mathbf{SE}_{+1\ or\ +1\sim7}$, $\mathbf{MP}_{+1\ or\ +1\sim7}$, $\mathbf{MD}_{+1\ or\ +1\sim7}$, $\mathbf{AP}_{-27\sim0}$)

Fig. 6. The architecture of SB-LSTM model

D. Day-ahead and Week-ahead Forecasting Results

The examples of forecasting results of day-ahead and week-ahead load profiles are shown in Figs 7 to 9 for the test set of the minimum demand week, the average demand week and the maximum demand week. For the day-ahead forecasting, the weekly results are obtained as day-by-day predictions (one day after another) for Monday to Sunday in the same weeks, while for the week-ahead forecasting, the SB-LSTM model gives the predictions for the whole week in one forecasted sequence. Tables I to III quantify the performance of two SB-LSTM models through the comparisons of the corresponding MAE and mean absolute percentage error (MAPE) [25]. The absolute and percentage errors in the total, over-estimated and under-estimated energy consumptions (denoted as $\mathrm{E_T}$, $\mathrm{E_O}$ and $\mathrm{E_U}$, respectively) are also compared with the actual demands, assuming that the mean demand values are maintained over each averaging window of 30 minutes.

From the presented results, it can be seen that SB-LSTM method can provide accurate results for both day-ahead and week-ahead load forecasting, although the error is greater in the latter case. As illustrated in the previous sub-section, the generalization of day-ahead forecasting SB-LSTM model and the generalization of week-ahead forecasting SB-LSTM model are quite similar, which means that SB-LSTM approach is very flexible and it can capture deep dependencies and features of the analyzed data sequences.

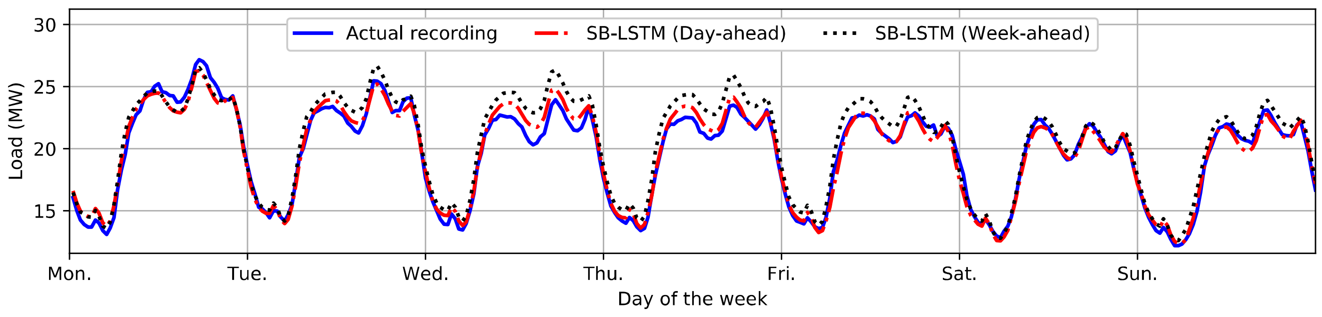

Fig. 7. Comparison of day-ahead and week-ahead load forecasting results for the week of minimum demand

Authorized licensed use limited to: Indian Institute of Technology (Ropar). Downloaded on May 23,2020 at 12:20:19 UTC from IEEE Xplore. Restrictions apply.

TABLE I. EVALUATION OF SB-LSTM PERFORMANCE (RESULTS IN FIG. 7, THE MINIMUM DEMAND WEEK)

Forecasting in the Minimum Demand Week	MAE (MW)	MAPE (%)	E_O (MWh)	E_O (%)	E_U (MWh)	E_U (%)	E_T (MWh)	E_T (%)	Total Demand (MWh)
Day-ahead	0.448	2.296	45.073	2.402	-30.143	-2.100	14.930	0.451	3311.580
Week-ahead	0.933	4.721	147.168	5.163	-9.505	-2.060	137.663	4.157	3311.580

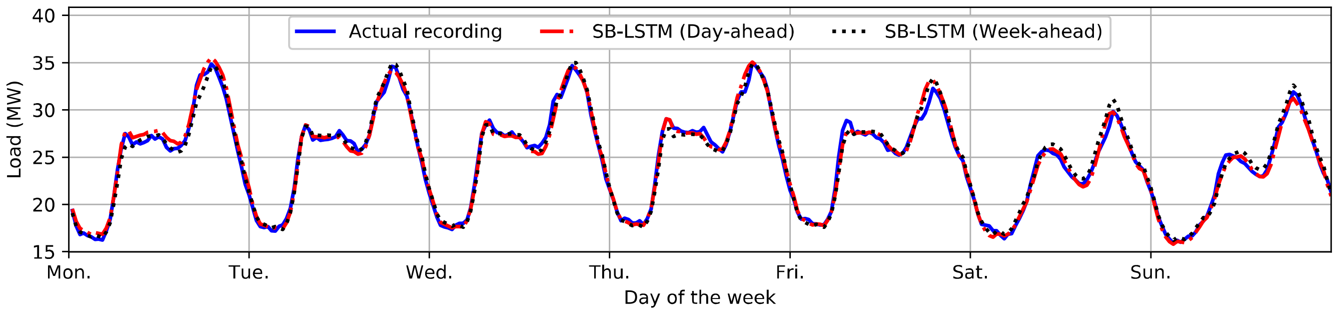

Fig. 8. Comparison of day-ahead and week-ahead load forecasting results for the week of average demand

TABLE II. EVALUATION OF SB-LSTM PERFORMANCE (RESULTS IN FIG. 8, THE AVERAGE DEMAND WEEK)

Forecasting in the Average Demand Week	MAE (MW)	MAPE (%)	E_O (MWh)	E_O (%)	E_U (MWh)	E_U (%)	E_T (MWh)	E_T (%)	Total Demand (MWh)
Day-ahead	0.411	1.635	38.432	1.717	-30.688	-1.583	7.744	0.185	4176.947
Week-ahead	0.495	1.973	50.619	1.963	-32.622	-2.041	17.998	0.431	4176.947

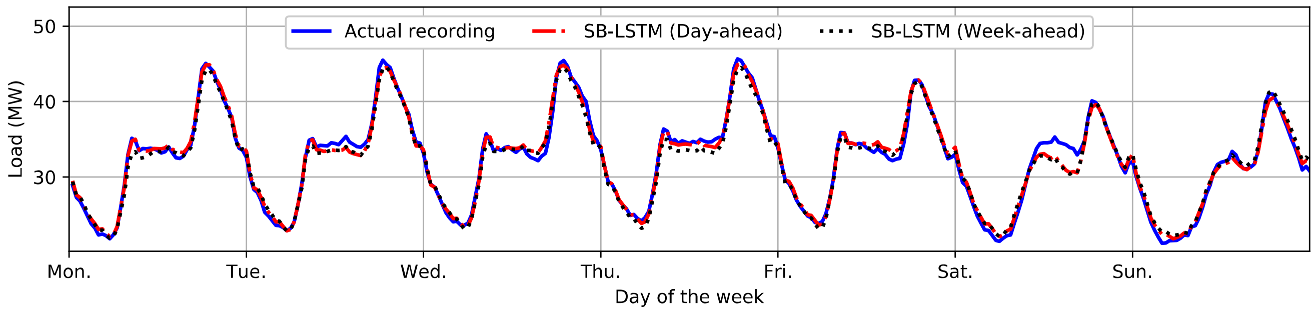

Fig. 9. Comparison of day-ahead and week-ahead load forecasting results for the week of maximum demand

TABLE III. EVALUATION OF SB-LSTM PERFORMANCE (RESULTS IN FIG. 9, THE MAXIMUM DEMAND WEEK)

Forecasting in the Maximum Demand Week	MAE (MW)	MAPE (%)	E_O (MWh)	E_O (%)	E_U (MWh)	E_U (%)	E_T (MWh)	E_T (%)	Total Demand (MWh)
Day-ahead	0.525	1.660	44.182	1.553	-44.028	-1.695	0.154	0.003	5442.538
Week-ahead	0.708	2.218	41.718	2.050	-77.274	-2.267	-35.555	-0.653	5442.538

IV. CONCLUSIONS

This paper provides comparison of day-ahead and week-ahead load forecasting with SB-LSTM approach, based on available historical demand recordings and weather (meteorological) data. The SB-LSTM method is a type of deep recurrent neural network model, which is capable of resolving the gradient vanishing and gradient exploding issues of the traditional recurrent neural network models. In SB-LSTM method, the input can naturally preserve the full correlational information on temporal dependencies. For load forecasting problems, SB-LSTM analyzes all load recordings and all meteorological forecast information at all moments and their hidden correlations and deep patterns, providing robust and promising predictions.

Unlike classical artificial feedforward neural network, which is relatively straightforward and naïve, and in which input signals can only travel in one direction to the output layer with no feedback and memory structure, the presented recurrent neural network SB-LSTM method is implemented with "memory mechanism", represented as a recurring connection to itself. In that way, SB-LSTM method can efficiently "learn" about the relevant distinctive temporal features of the whole input data sequence.

TABLE IV. COMPARISON OF COMPUTATIONAL PERFORMANCE

Method		Memory Required	Computational Time
SB-LSTM	***Day-Ahead***	4.6 GB	Typically 90 min
	Week-Ahead	5.2 GB	Typically 120 min

Authorized licensed use limited to: Indian Institute of Technology (Ropar). Downloaded on May 23,2020 at 12:20:19 UTC from IEEE Xplore. Restrictions apply.

Table IV gives a brief comparison of computational requirements. The results were obtained by running the training sections several times to reduce the uncertainties and by taking the average value of these runs. All trainings are performed on the same hardware: AMD Ryzen 1800X (SMT disabled, 8 cores @ 4.015 GHz), 16 GB RAM @ 3200 MHz (C14), Nvidia GTX 1080ti (GPU clock @ 1987 MHz). The executing platform was MATLAB R2019a. It can be seen that memory requirements are modest and that trainings can be finished within relatively short time periods (around two hours for week-ahead case).

As expected, the presented results show that day-ahead load forecasting has a higher accuracy than week-ahead forecasting, but there are days, or parts of days, where both day-ahead and week-ahead forecasting fail to provide predictions with high accuracies. The load profiles in these days are significantly different from the other days that were used during the learning/training process by the models. The main reasons for these differences may be due to the impact of "hidden" (i.e. not considered) variables, e.g. planned or unplanned network maintenance, possible demand-side-management actions and similar, which all increase complexity and bring higher uncertainty to the network operating conditions, making resulting forecasting less accurate than in the case of "standard demand days".

To further improve the performance of the presented SB-LSTM method, there are three "candidate approaches": 1) increasing the size of the processed dataset, to allow capturing of variations not included in the limited available dataset; 2) implementing more advanced architectures, where, for example, attention mechanism can be used to improve the accuracy of LSTM-based models by assigning various weights to the input elements; and 3) providing forecasted demands with indicated uncertainties or confidence levels, e.g., by implementing probabilistic forecasting, which can be added in the presented models by using Bayesian learning algorithm. This is subject of the further work by the authors.

References

[1] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-Term Residential Load Forecasting Based on LSTM Recurrent Neural Network," *IEEE Trans. on Smart Grid,* vol. 10, no. 1, pp. 841-851, 2017.

[2] R. Weron, *Modeling and Forecasting Electricity Loads and Prices: A Statistical Approach*. Wiley Finance Series, 2006.

[3] G. Xie, X. Chen, and Y. Weng, "An Integrated Gaussian Process Modeling Framework for Residential Load Prediction," *IEEE Transactions on Power Systems,* vol. 33, no. 6, pp. 7238-7248, 2018.

[4] K. Y. Lee, Y. T. Cha, and J. H. Park, "Short-term Load Forecasting Using an Artificial Neural Network," *IEEE Trans. on Power Systems,* vol. 7, no. 1, pp. 124-132, 1992.

[5] S. Kyung-Bin, B. Young-Sik, H. Dug Hun, and G. Jang, "Short-term Load Forecasting for the Holidays Using Fuzzy Linear Regression Method," *IEEE Trans. on Power Systems,* vol. 20, no. 1, pp. 96-101, 2005.

[6] H. Guo, "Accelerated Continuous Conditional Random Fields For Load Forecasting," *IEEE Trans. on Knowledge and Data Engineering,* vol. 27, no. 8, pp. 2023-2033, 2015.

[7] E. Ceperic, V. Ceperic, and A. Baric, "A Strategy for Short-Term Load Forecasting by Support Vector Regression Machines," *IEEE Trans. on Power Systems,* vol. 28, no. 4, pp. 4356-4364, 2013.

[8] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comp.,* vol. 9, no. 8, pp. 1735-1780, 1997.

[9] G. Chen, "A Gentle Tutorial of Recurrent Neural Network with Error Backpropagation," arXiv, 2016. [Online]. Available: https://arxiv.org/pdf/1610.02583.pdf

[10] M. C. Mozer, "A Focused Backpropagation Algorithm for Temporal Pattern Recognition," in *Backpropagation: Theory, Architectures and Applications*. L. Erlbaum Associates Inc., 1995, pp. 137-169.

[11] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Trans. on Signal Processing,* vol. 45, no. 11, pp. 2673-2681, 1997.

[12] M. Hermans and B. Schrauwen, "Training and Analyzing Deep Recurrent Neural Networks" 26th Int. Conf. on Neural Information Processing Systems - Volume 1, Lake Tahoe, Nevada, 2013.

[13] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors," arXiv, 2012. [Online]. Available: https://arxiv.org/pdf/1207.0580.pdf

[14] A. Y. Ng, "Feature Selection, L1 vs L2 Regularization and Rotational Invariance," *21st Int. Conf. on Machine Learning, ICML '04,* 2004.

[15] A. Paisios, "Profiling and Disaggregation of Electricity Demands Measured in MV Distribution Networks," PhD thesis, The University of Edinburgh, 2017.

[16] "Met Office (2019): MIDAS Open: UK Hourly Weather Observation Data, v201901, "Centre for Environmental Data Analysis (CEDA), March 2019.

[17] C. A. Hall and W. W. Meyer, "Optimal Error Bounds for Cubic Spline Interpolation," *J. of Approx. Theory,* vol. 16, no. 2, pp. 105-122, 1976.

[18] "Scottish Bank Holidays", Scottish Government, 2019. [Online]. Available: https://www.gov.scot/publications/bank-holidays/

[19] J. Zhang, Y. Lai, and J. Lin, "The Day-of-the-Week Effects of Stock Markets in Different Countries," *Finance Res. Letters,* vol. 20, pp. 47-62, 2017.

[20] J. LeSage, "Analyzing the Sun's Path," MATLAB Central File Exchange, 2016. [Online]. Available: https://uk.mathworks.com/matlabcentral/fileexchange/55510-analyzing-the-sun-s-path

[21] M. Zimecki, "The Lunar Cycle: Effects on Human and Animal Behavior and Physiology," *Postępy Higieny i Medycyny Doświadczalnej (Advances in Hygiene and Experimental Medicine),* vol. 60, pp. 1-7, 2006.

[22] M. Mahooti, "Moon Phases," MATLAB Central File Exchange, 2019. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/55270-moon-phases

[23] D. Harris and S. Harris, *Digital Design and Computer Architecture, Second Edition*. Morgan Kaufmann Publishers Inc., 2012, p. 712.

[24] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Int. Conf. on Learning Repr.*, 2015.

[25] M. Shcherbakov, A. Brebels, N. L. Shcherbakova, A. Tyukov, T. A. Janovsky, and V. A. Kamaev, *A Survey of Forecast Error Measures*. 2013, pp. 171-176.

Authorized licensed use limited to: Indian Institute of Technology (Ropar). Downloaded on May 23,2020 at 12:20:19 UTC from IEEE Xplore. Restrictions apply.