LSTM-based Short-term Load Forecasting for Building Electricity Consumption

Xin Wang, Fang Fang,
Xiaoning Zhang, Yajuan Liu
School of Control & Computer Engineering,
North China Electric Power University,
Beijing, China,
15646522828@163.com.

Le Wei
Department of Automation,
North China Electric Power University,
Hebei, China.

Yang Shi
Department of Mechanical Engineering,
University of Victoria,
British Columbia, Canada.

Abstract—The unprecedented level of flexibility in energy management is required to ensure the balance of real-time energy production and consumption. Accurate short-term load forecasting (STLF) is vital for making the intelligent operation scheme. However, conventional forecasting techniques may not meet the increasingly demanding precision in load forecasting. This paper presents a novel energy load forecasting methodology based on Recurrent Neural Network (RNN), specifically Long Short-term Memory (LSTM) algorithms. The proposed LSTM-based model was trained and tested on a benchmark dataset which contained electricity consumption data for different kinds of buildings in America with one-hour resolution. The comparative models including multi-layer perceptron neural network (MLP), random forest (RF), and kernelized support vector machine (SVM) was also tested on the same dataset. The week-ahead forecasting results have shown that the proposed LSTM-based model outdoes the three comparative models in nine of twelve months.

Index Terms—Building Energy; Short-term Load Forecasting; Long Short-term Memory; Recurrent Neural Network

I. Introduction

The energy consumption of buildings accounts for a significant proportion of the whole energy consumption worldwide [1]. As a major energy consumer, buildings are mainly responsible for energy wastage as well. Thus it is necessary and urgent to introduce accurate energy consumption forecasting to make the energy utility of buildings more efficient.

Load forecasting can be divided into three categories according to the forecast horizon: 1) short-term load forecasting (STLF), 2) mid-term load forecasting, and 3) long-term load forecasting. Most of the current works on load forecasting have been focused on STLF, the forecast horizon of which does not exceed two weeks.

The reasons for the popularity of STLF lie in many aspects. For the grid, accurate STLF is crucial for efficient real-time power generation, distribution and dispatching. Nowadays, many renewable energy sources which are full of volatility and stochasticity have been incorporated into smart grids while the unprecedented flexibility is required to adapt to the changes in demand dynamically. Therefore, the accurate short-term forecasting of the demand will be propitious to devising suitable schemes to balance the energy production and consumption in real time. For the buildings, STLF is dramatically effective for mitigating uncertainties of the future electricity consumption [2]. The electricity consumption of one building is affected by many factors including weather condition, building palisade structure, the schedules of devices in the building, etc [3]. If STLF can take these factors into account, forecasting results will be obtained. The uncertainty will be weakened to contribute to carrying out demand response.

Two main methods can be found for handling load forecasting: 1) physics principle based models and 2) data-driven models. The current works are mainly focused on data-driven models. Accurate inputs are required by physics principle based models, otherwise it can result in poor forecasting. On the other hand, data-driven models do not require such detailed data about the studied building and learn from historical data for forecasting [1]. In [4], Ozgur Cemal Ozerdem et al. have proposed particle swarm optimized (PSO) artificial neural network (ANN) to predict hourly load supplied by an energy company. ANNs have been employed and modeled elaborately for all three categories of load forecasting in [5] and [6]. In [7], support vector machine (SVM) optimized by Cuckoo search (CS) algorithm for the filtered electricity load datasets has been proposed by Xiaobo Zhang et al. In [8], the electricity load demand datasets from Australian Energy Market Operator are first decomposed into several intrinsic mode functions (IMFs) by Empirical Mode Decomposition (EMD) algorithm. Then the extracted IMFs are predicted by a Deep Belief Network (DBN) including two restricted Boltzmann machines (RBMs) to obtain the final aggregated output. For simplicity, not all methods mentioned in the literature are introduced in this paper. For more details about different methods used for load forecasting, refer to [9] and [10].

As an effective improvement of recurrent neural network (RNN), LSTM has proven to be successful in load

The work is supported in part by the National Natural Science Foundation of China under Grant 51676068, and in part by the Beijing Municipal Science and Technology Project under Grant Z181100005118005.

978-1-7281-3666-0/19/$31.00 ©2019 IEEE

forecasting. Self-loops are introduced to produce paths where the gradient can flow in order to make continuous abstract for the long sequence. In this paper, the structure and training process of the proposed LSTM-based model have been presented in detail. The proposed model is tested on a benchmark dataset which contained electricity consumption data for different kinds of buildings in America with one-hour resolution. To evaluate the performance of the proposed LSTM-based model, multilayer perceptron neural network (MLP), random forest (RF), and kernelized SVM are also tested on the same dataset. The performance of the proposed model outdo the ones produced by the comparative models in nine of twelve months.

II. Long Short-term Memory

LSTM is an efficient variation of RNN. To be specific, researchers always encounters the vanishing/exploding gradient problem using a simple RNN, which prevents the loss function of RNN from converging to the minimal value.

A simple example of RNN is depicted as shown in fig. 1 to delineate the problems. For simplicity, the input vector is $[1, 0, \ldots, 0]$ with 999 zeros. The initial state is 0. The input weights and output weights are all 1s, and the linear function is employed as the activation function of the hidden layer. Considering all the given conditions, the final output o_{1000} is equal to w^{999}. Thus if w is only a little bigger or smaller than 1, the gradient of outputs will be either too large or too small.

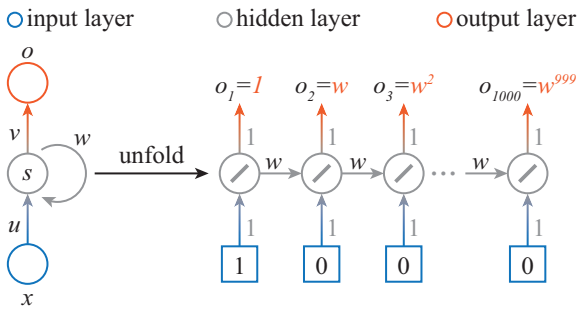

Fig. 1. A simple example of the vanishing/exploding gradient problems of RNN

To solve the vanishing gradient problem and incorporate long-term dependencies, Sepp Hochreiter & Juergen Schmidhuber proposed LSTM in 1997 [11]. Compared to the naive neural in a simple RNN, the LSTM cell is quite capable of deciding the degree of how information is accepted, stored and outputted, as shown in Fig. 2. These operations are completed by three flexible gates respectively: 1) input gate, 2) forget gate, and 3) output gate.

The three gates' opening is computed by the sigmoid function $\sigma(\cdot)$, whose inputs are weighted i_t and o_{t-1}. Each gate outputs a number between 0 and 1 for each scalar in weighted input vectors. 1 represents "fully open" and 0 represents "fully closed". Firstly, the weighted i_t and o_{t-1} are transformed by a tanh function to obtain the update signal u. Then, u and previous cell state s_{t-1} go through the input gate and forget gate respectively by element-wise multiplication. After that, the cell state gets updated to s_t by adding above gates' outputs. Lastly, the output o_t is computed by element-wise multiplication of transformed s_t and output gate opening. The overall process can be defined from (1) to (6):

$$i_g = \sigma(W_i(o_{t-1}, i_t) + b_i) \tag{1}$$

$$f_g = \sigma(W_f(o_{t-1}, i_t) + b_f) \tag{2}$$

$$o_g = \sigma(W_o(o_{t-1}, i_t) + b_o) \tag{3}$$

$$u = \phi(W_u(o_{t-1}, i_t) + b_u) \tag{4}$$

$$s_t = f_g \circ s_{t-1} + i_g \circ u \tag{5}$$

$$o_t = o_g \circ \phi(s_t) \tag{6}$$

where i_t is the input data at time step t. i_g, f_g and o_g refer to the input gate, forget gate and output gate respectively. u is the update signal. s_t is the cell state at time step t, o_t is the cell output at the same time. $\circ$ represents the element-wise multiplication. $\sigma(\cdot)$ is the sigmoid activation function, while $\phi(\cdot)$ is the tanh activation function.

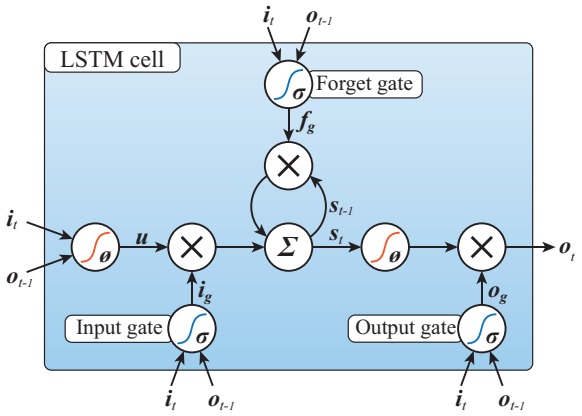

Fig. 2. Architecture of a LSTM cell

III. Methodology

This section is composed of three subsections. The first subsection introduces the provenance of electricity consumption datasets and analyses the corresponding weather datasets. The second subsection elaborates the proposed LSTM-based model for load forecasting. And the last subsection discusses the error metrics to quantitatively compare the performance of the proposed model and other models for comparison.

A. Datasets and Feature Selection

The proposed model was implemented on the datasets of electricity consumption for a large office located in Alaska, the United States, which was simulated by EnergyPlus based on the Department of Energy (DOE) commercial reference building models. It contained aggregated electricity consumption measurements and five sub-categories metering. In this paper, only the aggregated electricity consumption data is used. For more information about the datasets, refer to [12].

It is unquestioned that the electricity consumption of a building is correlated with the weather condition it stands in. Hence this paper also employs the corresponding weather datasets named Typical Meteorological Year, version 3 (TMY3) datasets. The TMY3 datasets are composed of 12 typical meteorological months that are concatenated essentially without modification to form a single year with a serially complete data record for primary measurements. It contains as many as 68 fields. In order to pick out useful fields, the feature importance is plotted using Random Forest regressor. The result shows that dry-bulb temperature, dew-bulb temperature, Relative humidity, and wind speed are the four most important features.

Specifically, the electricity consumption of an office building is highly correlated with schedule variables because of the distinct consumption level between weekdays and weekends. Hence the schedule variables are also contained as a compulsory part of the inputs. It's notable that the real-time weather condition for forecasting cannot be obtained, one-step lagged ones are adopted in fact. The slice of the training set is shown as table I.

As stated above, the inputs i_t for the electricity consumption forecasting in the office building are comprised of weather variables and schedule variables:

- Weather variables(one-hour lagged): dry-bulb temperature, dew-bulb temperature, relative humidity, and wind speed.
- Schedule variables: hour of day(1-24) and day of week(0-6).

B. The LSTM-based Model for Load Forecasting

To achieve the best accuracy, the proposed LSTM-based model uses a five-layer network structure including an input layer, two LSTM layers, a normal dense layer, and a dense output layer without activation function. Fig. 3 depicts the architecture of the proposed model.

Unlike the inputs for a normal neural model, the LSTM-based model accepts a sequence of input vectors $[i_1, i_2, \ldots, i_m]$ every time for the sake of producing short-term memory. In this paper, the sequence length is set to 24 because the electricity consumption has a daily periodicity.

Several measurements are taken to improve the performance of the proposed model. The data preprocessing

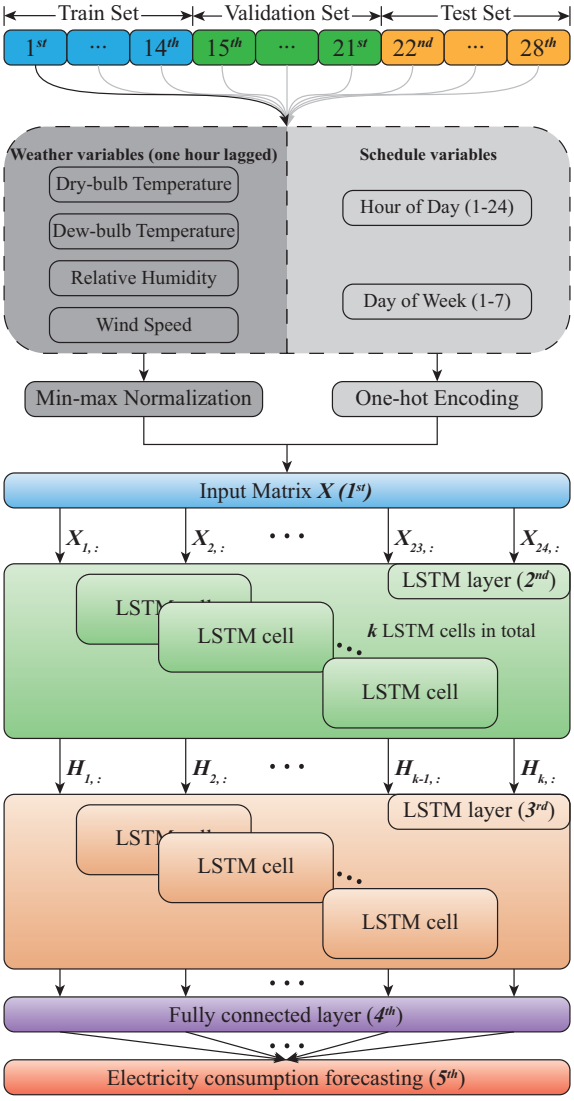

Fig. 3. Architecture of the LSTM-based model

is an important step before training. For the weather variables and electricity consumption data, the Min-max normalization is employed to scale values to $[0, 1]$, which can be expressed as (7):

$$X = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{7}$$

where x is the original value, $x_{\min}$ and $x_{\max}$ represent the minimum and maximum value respectively in the set of x. X is the normalized value.

For the schedule variables which are discrete, the one-hot encoding is adopted. This method encodes an original element which has M possible values to a vector with M elements, where only the corresponding new element is 1 while the rest new elements are all 0. For instance, Sunday (original value is 7) is encoded to a new vector $[0, 0, 0, 0, 0, 0, 1]$.

TABLE I
Slice of the Training Set

Time	Dry-bulb temperature (C°)	Dew-bulb temperature (C°)	Relative humidity (%)	Wind speed (m/s)	Hour of day (1-24)	day of week (0-6)
01-01 00:00	-14.4	-16.1	87	1.5	1	6
01-01 01:00	-15.0	-16.1	91	2.1	2	6
01-01 02:00	-14.4	-15.6	91	0.0	3	6
01-01 03:00	-13.9	-16.1	84	0.0	4	6
01-01 04:00	-13.9	-15.0	91	1.5	5	6
01-01 05:00	-15.0	-15.6	96	1.5	6	6
01-02 00:00	-15.0	-18.3	76	3.1	1	0
01-02 01:00	-15.0	-17.8	80	3.1	2	0
01-02 02:00	-15.0	-17.8	80	2.6	3	0
01-02 03:00	-15.0	-18.3	76	2.1	4	0
01-02 04:00	-15.6	-18.3	80	1.5	5	0
01-02 05:00	-15.6	-18.3	80	0.0	6	0

Dropout and weight decay regularization are two significant ways to avoid overfitting during training process. The former drops out a certain proportion of neural units in hidden layers at each training process stochastically. This curbs the co-dependence among neural units powerfully and contributes significantly to learning more robust characteristics. The latter is defined as multiplying weight in the gradient descent at each epoch by the parameter λ between 0 and 1. The regularization penalizes the weight in the neural network to relieve overfitting. It can be realized by adding a term to the cost function as:

$$C_2 = C_0 + \frac{\lambda}{2n}\sum_i \omega^2 \tag{8}$$

where C_0 is the original cost function. λ is a parameter controlling the degree of penalizing large weights. ω is the weight vector containing all free parameters of the network, and n is the length of ω.

C. Performance Metrics

The performance of the proposed model, MLP, RF, and kernelized SVM were evaluated quantitatively using mean absolute percentage error (MAPE). MAPE is a relative indicator which can eliminate the impact of the magnitude. MAPE is defined by Equation (9):

$$e_{\text{MAPE}} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \times 100\% \tag{9}$$

where y_t is the actual value and $\hat{y}_t$ is the forecasted value, n is the number of observations in the test set.

IV. Results

The proposed LSTM-based model was implemented using the Keras API which is running on the Tensorflow backend. For the application of week-ahead forecasting, the model used the electricity consumption of the first two weeks for training, the third week for validation, and the fourth week for testing. Mean absolute error (MAE) was used as the cost function. Adam [13] algorithm was adopted as the gradient-based optimizer instead of Stochastic Gradient Descent (SGD) for training in that Adam outperformed SGD in terms of faster convergence and lower error ratios. It's worth noting that the order of training and validation datasets couldn't be shuffled as usual because the LSTM-based model updated its memory based on the time series.

A. Hyper-parameters Optimization

Hyper-parameters are predetermined and cannot be optimized by training the model. The selection of hyper-parameters plays a significant role in the performance of the structured neural networks. The hyper-parameters in this model include: the number of neurons in the 2^{nd}, 3^{rd}, and 4^{th} layer, the dropout in the 2^{nd}, 3^{rd}, and 4^{th} layer, the activation function in the 4^{th} layer, and the learning rate.

There are two common methods for hyper-parameters optimization: grid search and random search. The former is effective but slow at searching the whole search space, while the latter is fast, but could miss important points in the search space. They all have distinct shortcomings which made them less practical. In this paper, Bayesian optimization is employed to make a compromise of above methods. It updates prior information continuously by means of gaussian process. Thus much fewer iterations than grid search are required in Bayesian optimization. It's also robust for non-convex programming problems. As one implementation of Bayesian optimization, Hyperopt [14] was used in this paper to optimize the above hyper-parameters globally. It's an efficient python library providing algorithms and parallel architecture for hyper-parameters optimization. There are four main steps to use Hyperopt:

- Defining an objective function.
- Determining the search space.
- Specifying trail datasets.
- Selecting the search algorithm.

For the proposed model, mean squared error (MSE) of validation datasets was used as the objective function to evaluate a set of hyper-parameters founded in the search space. Table II presents the whole search space for hyper-parameters mentioned above. Fig. 4 shows the optimal search result of the hyper-parameters for the proposed model, with the red triangles indicating the best parameters corresponding to the minimal validation loss.

TABLE II
Search space for the LSTM-based model

Hyper-parameter	Range
number of neurons in the 2^{nd} and 3^{rd} layer	$[5, 25]$
number of neurons in the 4^{th} layer	$[16, 40]$
dropout in the 2^{nd}, 3^{rd}, and 4^{th} layer	$[0, 0.5]$
activation function in the 4^{th} layer	[ReLU, Sigmoid]
learning rate	$[10^{-2}, 10^{-3}, 10^{-4}]$

MLP used in this paper had two hidden layers. the number of units, dropout, and activation in every layer need to be determined. SVM have three import hyper-parameters: kernel type, γ, and C. Kernel type determines how to map the original data to high-dimensional space. γ belong to radial basis function (RBF) which is a frequently-used kernel function in SVM. RF is an ensemble model of several decision trees. The number of decision trees is determined by the n-estimators parameter. For every decision tree, max depth and max features restrict the maximum depth and features respectively.

In order to compare the proposed LSTM-based model with MLP, RF, and SVM more objectively, parameters of these models were also optimized by Hyperopt respectively. KNN and SVM took the negative cross-validation score as the objective function while MLP used the same objective function as in the LSTM-based model. Table III presents optimized parameters of aforementioned models.

B. Experimental Results

For the application of week-ahead load forecasting, the proposed model as well as MLP, SVM, and RF had been used to forecast the electricity consumption of the fourth week in May (between May 22 and May 28). Predictions of the electricity consumption using the aforementioned models are depicted in Fig. 5. The absolute error between the prediction and real electricity consumption is also plotted in Fig. 5 for analysis. It's clear that the proposed LSTM-based model has the highest prediction accuracy. SVM is almost as accurate as the proposed model in workdays, but when it comes to weekdays, SVM becomes a little bit worse compared to the proposed model. MLP suffers from the same shortcomings with SVM and is worse than SVM in weekdays. The prediction of RF has intense fluctuation whether on workdays or on weekdays.

For a deeper and comprehensive comparison, the MAPE of the four models on test datasets in all twelve months is shown as fig 6. Generally, the performance of the proposed LSTM-based model is better than the comparative models in nine months. To be specific, the MAPE of the proposed LSTM-based model is just around 10% mostly and the minimum MAPE happened in April is 3%. SVM and MLP worse than the proposed LSTM-based model while RF are not acceptable. The reason for the relatively unsatisfied performance of RF is it can not output values continuously when used for regression.

TABLE III
Optimized hyper-parameters for all models

Model	Hyper-parameter	Value
LSTM	2^{nd} layer units	20
	3^{rd} layer units	5
	4^{th} layer units	20
	2^{nd} layer dropout	0.40
	3^{rd} layer dropout	0.10
	4^{th} layer units	0.16
	4^{th} layer activation	ReLU
	learning rate	10^{-2}
MLP	2^{nd} layer units	34
	3^{rd} layer units	27
	2^{nd} layer dropout	0
	3^{rd} layer dropout	0.15
	2^{nd} layer activation	ReLU
	3^{rd} layer activation	Sigmoid
	learning rate	10^{-2}
RF	n estimators	13
	max depth	19
	max features	3
	criterion	MSE
SVM	C	2.97
	γ	0.23
	kernel type	RBF

V. Conclusion

The goal of the presented work in this paper is to improve the feasibility in using LSTM for building electricity consumption forecasting. This paper presents a LSTM-based model for STLF. The proposed LSTM-based model is trained and tested on a benchmark dataset which contained electricity consumption data for different kinds of buildings in America with one-hour resolution. In order to evaluate the proposed model relatively, MLP, RF, and SVM are also created and tested on the same dataset. The week-ahead forecasting results had show that the proposed LSTM-based model were able to forecast building electricity consumption better than the comparative models in nine of twelve months, the best MAPE of which is only 3%. The performance of the model presented in this study can be further improved by incorporating lagged electricity consumption data into the training dataset.

References

[1] K. Amasyali and N. M. El-Gohary, "A review of data-driven building energy consumption prediction studies," Renewable and Sustainable Energy Reviews, vol. 81, pp. 1192-1205, 2018.

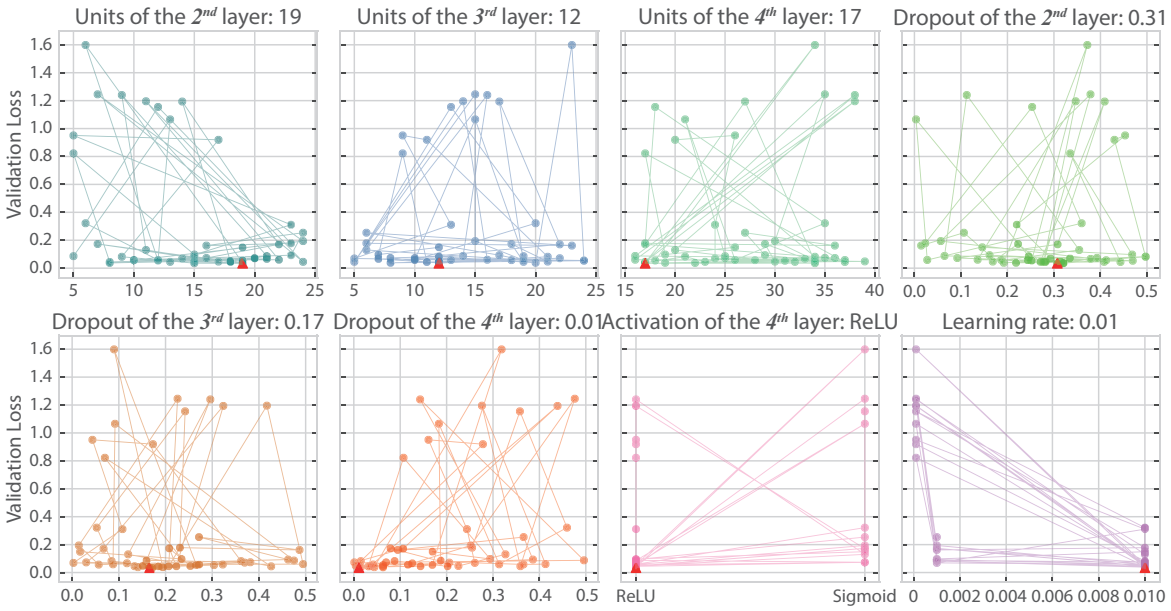

Fig. 4. Hyper-parameters optimization for LSTM-based model

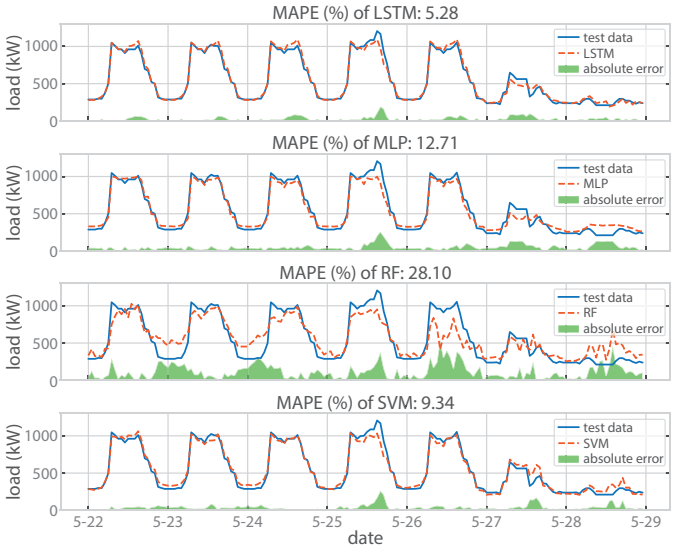

Fig. 5. Forecasting using the four models (May 22 - May 28)

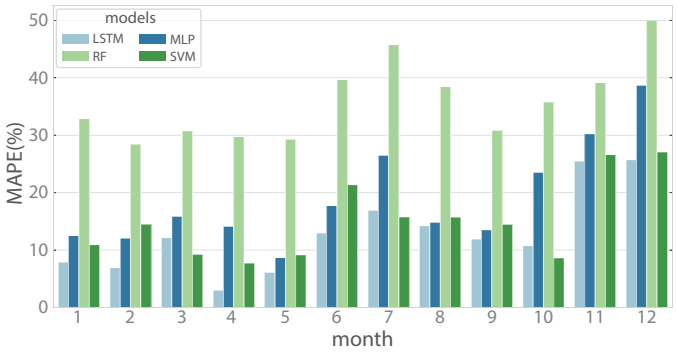

Fig. 6. MAPE of the four models in twelve months

[2] E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Deep learning for estimating building energy consumption," Sustainable Energy, Grids and Networks, vol. 6, pp. 91-99, 2016.

[3] J. Chen, X. Wang, and K. Steemers, "A statistical analysis of a residential energy consumption survey study in Hangzhou, China," Energy and Buildings, vol. 66, pp. 193-202, 2013.

[4] O. C. Ozerdem, E. O. Olaniyi, and O. K. Oyedotun, "Short term load forecasting using particle swarm optimization neural network," Procedia Computer Science, vol. 120, pp. 382-393, 2017.

[5] A. Baliyan, K. Gaurav, and S. K. Mishra, "A Review of Short Term Load Forecasting using Artificial Neural Network Models," Procedia Computer Science, vol. 48, pp. 121-125, 2015.

[6] C. Kuster, Y. Rezgui, and M. Mourshed, "Electrical load forecasting models: A critical systematic review," Sustainable Cities and Society, vol. 35, pp. 257-270, 2017.

[7] X. Zhang, J. Wang, and K. Zhang, "Short-term electric load forecasting based on singular spectrum analysis and support vector machine optimized by Cuckoo search algorithm," Electric Power Systems Research, vol. 146, pp. 270-285, 2017.

[8] X. Qiu, Y. Ren, P. N. Suganthan, and G. A. J. Amaratunga, "Empirical Mode Decomposition based ensemble deep learning for load demand time series forecasting," Applied Soft Computing, vol. 54, pp. 246-255, 2017.

[9] C. Deb, F. Zhang, J. Yang, S. E. Lee, and K. W. Shah, "A review on time series forecasting techniques for building energy consumption," Renewable and Sustainable Energy Reviews, vol. 74, pp. 902-924, 2017.

[10] B. Yildiz, J. I. Bilbao, and A. B. Sproul, "A review and analysis of regression and machine learning models on commercial building electricity load forecasting," Renewable and Sustainable Energy Reviews, vol. 73, pp. 1104-1122, 2017.

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

[12] Commercial and Residential Hourly Load Profiles for all TMY3 Locations in the United States. Office of Energy Efficiency & Renewable Energy (EERE). [Online]. Available: https://openei.org/datasets/files/961/pub/.

[13] D. P. Kingma and J. L. Ba, "ADAM: A Method for Stochastic Optimization," in ICLR, San Diego, 2015.

[14] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox, "Hyperopt: a Python library for model selection and hyperparameter optimization," vol. 8, no. 1, p. 014008, 2015.