Comparing high accurate regression models for short-term load forecasting in smart buildings

T T Q Nguyen, T P T Tran, Vincent Debusschere, Christophe Bobineau, Rémy Rigo-Mariani

Comparing high accurate regression models for short-term load forecasting in smart buildings

T.T.Q. Nguyen*, T.P.T. Tran*, Vincent Debusschere†, Christophe Bobineau*, and Rémy Rigo-Mariani†
*Univ. Grenoble Alpes, CNRS, Grenoble INP⋆, LIG, F-38000 Grenoble, France
†Univ. Grenoble Alpes, CNRS, Grenoble INP⋆, G2ELab, F-38000 Grenoble, France

***Abstract*—Load prediction is at the core of many smart grid topics because its accuracy affects the resilience and optimal operation of the power grid. Short term load forecasting (STLF) specific to buildings is needed for both demand and production sides management, to help reducing peak electricity demand, optimizing energy planning, management, and conservation among other. In this paper, we analyze and compare five of the best state of art performing regressive STLF algorithms with deep learning techniques. These methods are compared in the same context with the same real (large) data-set, leading to a fair comparison and precise evaluation. The load forecasting is deployed with and without taking into account environmental features. The idea is to help determine which one could be considered in the optimization loop of an energy management systems, either at the level of a district (local energy communities, or smart districts) or directly at the level of individual households (smart residential buildings).**

***Index Terms*—Regression model, Seq2Seq model, attention mechanism, Deep multilayer perceptron, Long short term memory, Support vector regression, random forest, short-term forecasting, load forecasting.**

I. Introduction

Smart grid technologies help collecting and analyzing a huge volume of data, with the objective of optimizing the management of the power system at both the producer and the demand sides. This volume of data can contribute to the optimization of the power usage, considering economic benefits as well as environmental constraints. The forecast of the loads consumption (for both tertiary, commercial and residential buildings) is at the core of most of the optimization tools design to take the good decision in power conservation and management planning [1].

Load forecasting can be divided into three categories according to the time scale [2].

1) **Short-term:** From a few minutes to a few days ahead;
2) **Medium-term:** From a few days to a few months ahead;
3) **Long-term:** From months to years.

Most of the researches has focused on *short-term load forecasting* (STLF), because it is critical for real-time power generation, operation and economic transactions. Besides, it is difficult to achieve an acceptable accuracy for short-term load forecasting. In fact, smart grids technologies intend to collect sensors data at a higher sample rate, targeting real-time monitoring to obtain an improvement in the prediction result as well. STLF is also complex because of the variety of the data-set. Thus, it is challenging, for the predicted model, to capture relevant variations. For instance, buildings energy consumption do not follow clear patterns, rules or cyclic elements. The output of the prediction algorithm (the power consumption) is a real number that is influenced by many input factors such as temperature, humidity, wind speed , most of which are exogenous and quite hard to forecast.

Two main approaches are considered currently in forecasting techniques: physical based and data-driven based ones. The physical based techniques, such as EnergyPlus or Ecotest [3], are based on thermodynamics relationships to build the energy model. The accuracy of the predicted values depends on the accuracy of the simulated model and its inputs. However, some details of data are not available in simulation time. This leads to significant uncertainty in the forecasting. On the contrary, the data-driven based technique do not require physical detailed models and the forecasting process is performed based on historical data only.

Plenty of regressive models exist in short-term building load consumption forecasting. They are developed in four main categories [1]:

1) *Support vector regression* (SVR);
2) *Artificial neural networks* (ANN): *Bidirectional fuzzy neural network* (BFNN), *feed forward neural network* (FFNN), *multilayer perceptron* (MLP), *long short-term memory* (LSTM);
3) *Decision tree* (DT): *Random forest* (RF), *classification and regression trees* (CART);
4) Statistics methods: *Multiple linear regression* (MLR), *auto-regressive integrated moving average* (ARIMA), *k-nearest neighbor* (kNN).

Statistic and SVR techniques are widely used to find out the most accurate models for short-term load forecasting. The computations are realized with real data-sets with hourly resolution, but the results are not usually optimistic. The performances of SVR techniques are generally better than statistic ones, as illustrated in [4], [5] for instance. ANN, such as FFNN and BFNN, are also widely implemented, though machine learning levels with one hidden layer only present an accuracy around 70 % [6].

Data-sets in STLF present a significant variation, which increases the uncertainty of the predicted values. In recent work, ANN techniques with deep learning models were showing promising results. However, their performances on benchmark

⋆ Institute of Engineering Univ. Grenoble Alpes

data was still to improve. Moreover, the size of the training data-set and the influencing features have to be adapted to such a deep learning based model for relevant results (not too small and if possible adaptive) [7].

In this paper, we focus on comparing and analyzing the most recent techniques with the highest accuracy. These methods are SVN, RF, MLP, LSTM and an attention-based LSTM. The objective is to identify the best model for the STLF of smart buildings and local energy communities (or smart districts) and evaluate factors impacting the accuracy of the prediction, with in mind a future usage of the data in an energy management system, i.e. based on an optimization loop. These models are trained on a real data-set with and without taking into account of weather conditions and time-related features.

The paper is structured as follow. First, the general theoretical models of the five techniques are presented. The analysis of the features impacting the predicted values and the choice of the parameters to train the models are explained in Section II. Then, the data-set used to compare the forecasting techniques is describe in Section III. Next, the discussion on the experimental results is presented in Section IV. Finally, the conclusions is drawn at the end of the paper.

II. Forecasting techniques

A. Definitions and performance metrics

The generic regression model for energy forecasting based on time series analytic is expressed in (1), where $\{\boldsymbol{x}_{t-T+1}, \ldots, \boldsymbol{x}_t\}$ is the feature of d-dimensional time series in the n^{th} moving window, including T time steps before $(t+1)$, and $\hat{y}_{t+1} \in \mathbb{R}$ is the prediction of the output values at time $(t+1)$.

$$\hat{y}_{t+1} = F(\boldsymbol{x}_{t-T+1}, \ldots, \boldsymbol{x}_t) \tag{1}$$

The objective function is to minimize the distance D between the true values y and the predictive values $\hat{y}$, as expressed in (2), where t can be any value or time period, and D is any distance.

$$\min_F \sum_t D(y_t; \hat{y}_t) \tag{2}$$

We evaluate the accuracy of the techniques with two criteria: the *mean absolute percentage error* (MAPE) and the *root mean squared error* (RMSE).

$$\text{MAPE} = \frac{1}{N} \sum_{i=1}^{N} \frac{|y_i - \hat{y}_i|}{y_i} \times 100\,\% \tag{3}$$

$$\text{RSME} = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}} \tag{4}$$

The MAPE determines the average percentage of error per prediction, defined in (3), and the RMSE focuses on the deviation between each predicted value and its corresponding true value, defined in (4), with N the number of elements of the time series.

B. Implemented algorithms

1) Multilayer perceptron: MLP is a model of feed-forward neural network that is composed of at least three layers: input, hidden and output layers. Information are moved only one direction, "forward", i.e. from the input to the output, as shown in Fig. 1.

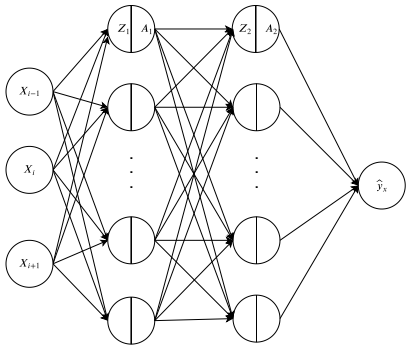


Fig. 1. Principle of the multilayer perceptron with two hidden layers.

The input layer is used to enter the input information, $\{\boldsymbol{x}_{t-T+1}, \ldots, \boldsymbol{x}_t\}$, thus the number of unit is a product of the number of features and the width of the window. The output layer is only a single unit: the output value y_{t+1}. In the hidden layers, the nodes rely on a nonlinear activation function (e.g. `relu`, `tanh` or `sigmod`) to map the weighted inputs to the output of nodes. A 2-hidden layers MLP is formulated in (5), where $\boldsymbol{W}$ corresponds to a weight matrix and $\boldsymbol{b}$ is a bias parameter vector.

$$\begin{cases} \boldsymbol{Z}_0 = \text{flatten}(\boldsymbol{x}_{t-T+1}, \ldots, \boldsymbol{x}_t) \\ \boldsymbol{Z}_1 = \boldsymbol{W}_0.\boldsymbol{Z}_0 + \boldsymbol{b}_0 \\ \boldsymbol{A}_1 = f_1(\boldsymbol{Z}_1) \\ \boldsymbol{Z}_2 = \boldsymbol{W}_1.\boldsymbol{Z}_1 + \boldsymbol{b}_1 \\ \boldsymbol{A}_2 = f_2(\boldsymbol{Z}_2) \\ \hat{y}_{t+1} = \boldsymbol{W}_2.A_2 + b_2 \end{cases} \tag{5}$$

"flatten" is an operator that creates a vector with one row from a matrix. Deep MLP implementations present more than one hidden layer, which is more advantageous to learn complex functions F [8]. However, the subsequent increase in parameters requires a larger data-set to train the model.

2) Long short-term memory: LSTM is an artificial *recurrent neural network* (RNN) in which the output from previous step is fed as input to the current step, relying namely on a "feedback" connection. This network is capable to learn long-term dependencies and remember information for prolonged periods of time with the help of a memory cell (i.e. the previous time step). The architecture of LSTM is a chain of cells in which the information can be removed or added using gates, as illustrated in Fig. 2.

For the t^{th} cell, the input includes the current input information $\boldsymbol{x}_t$, the previous information of the state of the cell $\boldsymbol{h}_{t-1}$ and the memory cell $\boldsymbol{c}_{t-1}$. The information flows through three gates: the update gate $\boldsymbol{\Gamma}_u$, the forget gates $\boldsymbol{\Gamma}_f$ (used to update

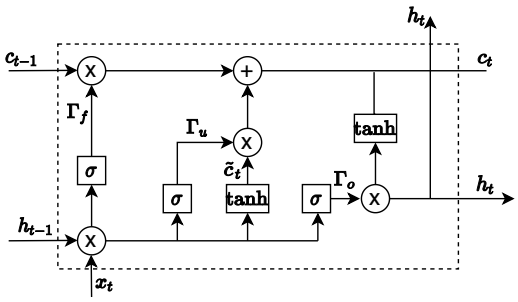


Fig. 2. Cells description of the long short-term memory implementation.

the cell memory $\boldsymbol{c}_t$) and the output gate $\boldsymbol{\Gamma}_o$ (used to update the state cell $\boldsymbol{h}_{t-1}$). The LSTM principle is defined in (6), where σ and tanh are the activation function.

$$\begin{cases} \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_c(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t) + \boldsymbol{b}_c) \\ \boldsymbol{\Gamma}_u = \sigma(\boldsymbol{W}_u(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t) + \boldsymbol{b}_u)) \\ \boldsymbol{\Gamma}_f = \sigma(\boldsymbol{W}_f(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t + \boldsymbol{b}_f)) \\ \boldsymbol{\Gamma}_o = \sigma(\boldsymbol{W}_o(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t + \boldsymbol{b}_o)) \\ \boldsymbol{c}_t = \boldsymbol{\Gamma}_u.\tilde{\boldsymbol{c}}_t + \boldsymbol{\Gamma}_f.\boldsymbol{c}_{t-1} \\ \boldsymbol{h}_t = \boldsymbol{\Gamma}_o.\tanh(\boldsymbol{c}_t) \\ \hat{y} = \boldsymbol{W}_s.\boldsymbol{h}_t + b_s \end{cases} \tag{6}$$

where $\boldsymbol{W}_c, \boldsymbol{W}_u, \boldsymbol{W}_f, \boldsymbol{W}_o, \boldsymbol{W}_s$ are the weight matrices parameters, $\boldsymbol{b}_c, \boldsymbol{b}_u, \boldsymbol{b}_f, \boldsymbol{b}_o, b_s$ are the bias vector parameters, and $\tilde{\boldsymbol{c}}_t$ is a new candidate of cell state.

LSTM is a special type of RNN that solves the problem of the vanishing gradient in the training process and that captures much more long-term dependencies, justifying its preferred selection in the last few years. Furthermore, deep LSTM, stacking at least two LSTM layers as presented in Fig. 3, addresses correctly the estimation of complex functions, as in our case load consumption in smart buildings.

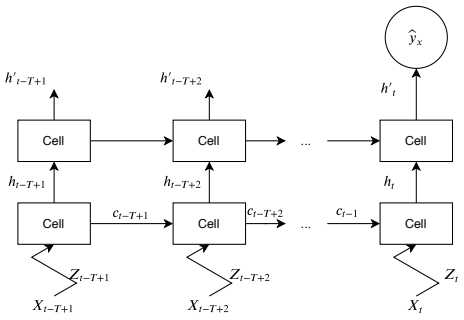


Fig. 3. Principle of the long short-term memory implementation with more than two layers.

3) Attention-based long short-term memory: At-LSTM is based on the attention mechanism, a part of the Seq2Seq model [9], which has demonstrated interesting qualities in multiple domains, particularly in language translation because of its capacity to capture important part of the input sequence. This improves the correlation between the input and the output data so that the model prediction is more accurate. Technically, the attention mechanism is applied to the top layer of an LSTM, as presented in Fig. 4.

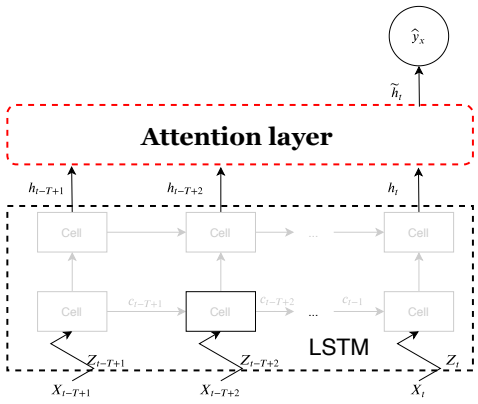


Fig. 4. Attention-based long short-term memory implementation.

The implementation of the At-LSTM is presented in (7), where $\boldsymbol{H}$ is the features extracted by the prediction model at the current step t, such as $\boldsymbol{H} = [\boldsymbol{h}_{t-T+1}, ..., \boldsymbol{h}_t]$; e is the content-based function of $\boldsymbol{H}$; $\boldsymbol{\alpha}$ denotes the attention weights of the features of $\boldsymbol{H}$; $\boldsymbol{r}$ is the context vector of the attention layer; $\tilde{\boldsymbol{h}}$ is the attention vector that produces the predicted output and V_α is a parameter learned with the rest of the system.

$$\begin{cases} \tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_c(\boldsymbol{r}_t, \boldsymbol{h}_t)) \\ \boldsymbol{r}_t = \sum_T^{i=1} \boldsymbol{\alpha}_{ti}.\boldsymbol{h}_i \\ \boldsymbol{\alpha}_{ti} = \frac{\exp(\boldsymbol{e}_{ti})}{\sum_n^{i=1} \exp(\boldsymbol{e}_{ti})} \\ \boldsymbol{e}_{ti} = \boldsymbol{V}_a^T \tanh(W_a(\boldsymbol{h}_t, \boldsymbol{H})) \end{cases} \tag{7}$$

4) Random forest: RF represents a type of learning methods that combine prediction from multiple models to increase accuracy and control over-fitting. RF techniques aggregate many DT, using bagging. An individual tree in this technique is built on a random subset of the data-set, with random features that can be split on at each node, as presented in Fig. 5.

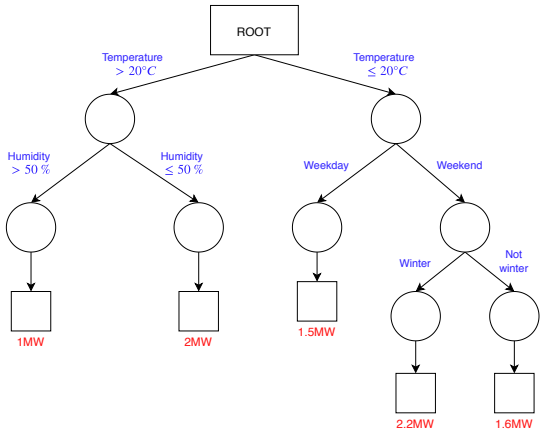


Fig. 5. Illustration of a sample random forest tree.

The average prediction from all the individual regression trees helps improving the predictive accuracy. Hence, RF is efficient on large data-sets.

5) Support vector regression: SVR positions an optimal hyperplane by maximizing, under constraints, the distance between the predicted value $\hat{\boldsymbol{y}}_{t+1}$ and the desired output $\boldsymbol{y}_{t+1}$. The principle of this method is illustrated in Fig. 6 and expressed in (8), where $\boldsymbol{w}$ is the normal vector of the hyperplane weight vector, ε a margin of error, and b a bias parameter.

$$\min \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 \text{ such that}$$
$$\mid y_{t+1} - \langle \boldsymbol{w}, \text{flatten}(\boldsymbol{x}_{t-T+1}, \ldots, \boldsymbol{x}_t) \rangle - b \mid \leq \varepsilon, \forall t \quad (8)$$

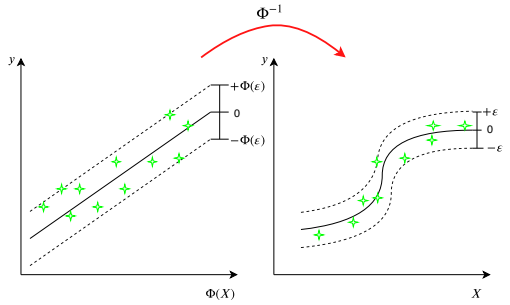


Fig. 6. Principle of the Support vector regression technique.

The standard SVR implementation is linear. Using alternative kernel functions (like the *polynomial* (poly), the *Gaussian radial basis function* (RBF), or the *Sigmoid*) to replace the inner product $\langle ., . \rangle$ in (8), creates a nonlinear SVR, much more convenient to analysis complex data. Furthermore, a soft margin can be employed by adding slack variables $\varepsilon_t^*, \varepsilon_t \geq 0$ to relax the optimization as expressed in (9), where Φ is a kernel function and C is a regularization parameter.

$$\min \frac{1}{2} \parallel \boldsymbol{w} \parallel^2 + C \sum_t \varepsilon_t^* + \varepsilon_t \text{ such that}$$
$$\begin{cases} y_{t+1} - \Phi(\boldsymbol{w}, \text{flatten}(\boldsymbol{x}_{t-T+1}, \ldots, \boldsymbol{x}_t)) - b \leq \varepsilon_t + \varepsilon_t^* \\ \Phi(\boldsymbol{w}, \text{flatten}(\boldsymbol{x}_{t-T+1}, \ldots, \boldsymbol{x}_t)) + b - y_{t+1} \leq \varepsilon_t + \varepsilon_t^* \end{cases} \quad (9)$$

III. Data-set

Many parameters influence the energy consumption, such as the ambient dry bulb temperature, humidity, solar radiance, dew point, cloud cover, wind speed, pressure or scheduling (weekdays/weekend/holidays) and so on [1]. Considering influencing factors with the energy consumption leads to an increase of the training data-set size, and then elongate the training time. Energy consumption trend is drawn from the same figure with these parameters. Discarding factors presenting lower or identical correlation coefficients (multi-co-linear) is recommended at this step.

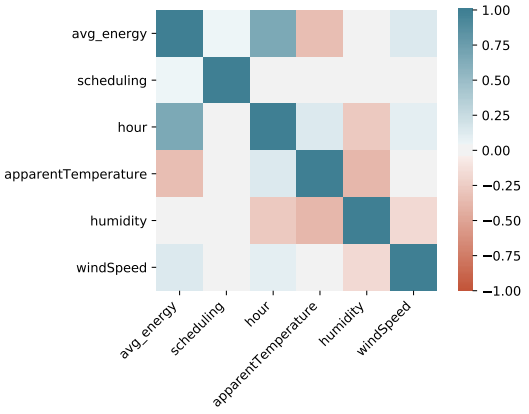


Fig. 7. Correlations of weather parameters and energy in the data-set [10].

Energy consumption data, collected from smart meters of 5 567 households in London, and historical weather data during more than 2 year, from November 2012 to February 2014, are available in Kaggle [10]. The dew point and UV index are correlated with the temperature and pressure, and the moon phase present insignificant correlation with energy, so they are discarded. Thus, the exogenous parameters are reduced to the temperature, the wind speed, the humidity and time scheduling data.

The correlation of weather parameters is shown in Fig. 7. The temperature and humidity present an inverse correlation to energy. The wind speed presents a low correlation with energy. We also consider two features for the time scheduling: the hour of the day (from 0 to 23) and the day of the week (0 for weekday or 1 for holiday and weekend).

IV. Results and Discussion

A. Scenarios and modeling

The prediction of hourly energy consumption is performed following four scenarios:

1) All buildings without influencing parameters;
2) All buildings with influencing parameters;
3) One private house without influencing parameters;
4) One private house with influencing parameters.

The idea is to compare the performances of the algorithms facing aggregated energy consumption patterns (for instance at the level of a local energy community or a smart district) or directly raw energy consumption curves, with much more variability (at the level of a smart household). Also, considering influencing parameters impacts the needed sensors information, thus indirectly the practical acceptability, replicability, and in the end the installation cost as well as the computational tractability. Those considerations define effectively four scenarios for the comparison, knowing that the usage of the forecasting algorithm lies within an optimization loop of an energy management system (with technical and economic constraints). This optimization layer will probably influence the performance metrics, without being directly considered in this paper.

Training data is aimed to predict the energy for the week-ahead. The energy consumption readings from the 1^{st} to the 7^{th} of January 2014 are used as test data, the rest is used for training and validation. Note that this week is randomly chosen. Indeed, for comparison between the technologies, we should focus on the manipulations on the same data-set, independently of it it is winter or summer.

Regression neural network (MLP and LSTM) are implemented with two neutron layers, each one built of 32 units. The *mean absolute error* (MAE) is used as loss function because its gradient varies significantly more than the *mean square error* (MSE) throughout iterations, as shown in Fig. 8.

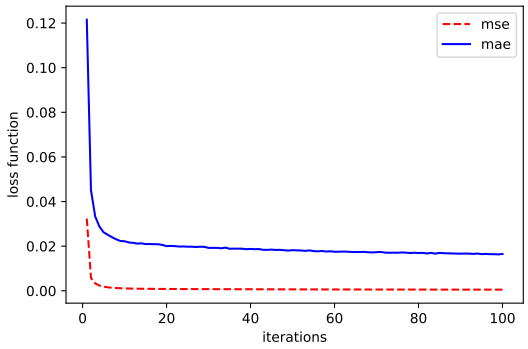


Fig. 8. Gradient of the mean square error (MSE) and the mean absolute error (MAE) as a function of the number of the number of iterations.

The Adam algorithm was adopted as the gradient-based optimizer because of its fast convergence rate and lower error ratios. Besides, the mean absolute values are steadily decreasing during training periods, so using dropout layer was ignored. The learning rate is automatically set by Adam.

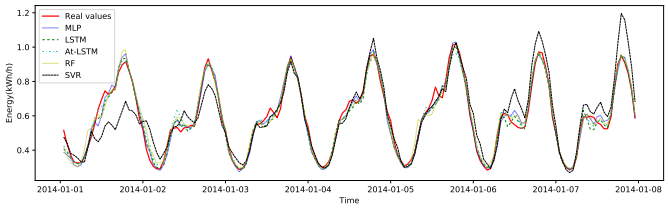
A single layer LSTM was used for the At-LSTM technique to better compare with the other techniques as well as to insure the best results for this paper.

The RF was implemented to create trees using random selections of features and samples. The `random_state` parameter allows controlling these random choices. It was fixed throughout the cases study to compare adequately the results after the executions. The SVR used a 4^{th} order polynomial function as kernel. Data in the past 24 hours (T=24) was used to predict one hour-ahead of energy consumption. As supplementary data, the detailed implementations of the four scenarios are presented in [11].

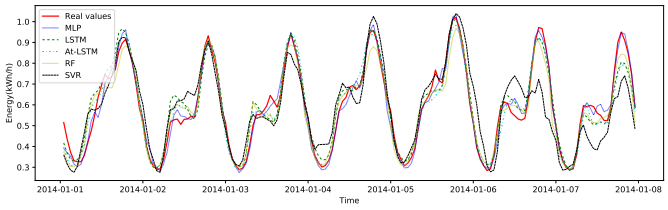
B. Comparison of the implemented techniques

The result of the prediction of the five implemented techniques in the case of the hourly average load consumption prediction of all buildings and only a single private household are shown in Fig. 9.

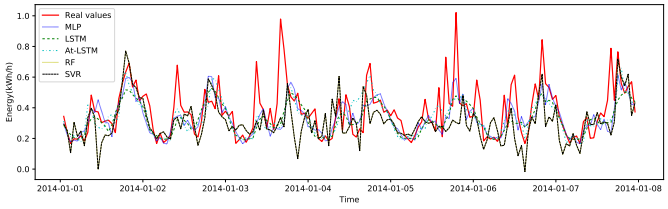
With the same data-set, the accuracy of the average energy consumption prediction error MAPE is high for the whole set of houses, reaching up to 96 %. Meanwhile, the prediction for the private house is less accurate, simply due to the increased variability of the load curve.



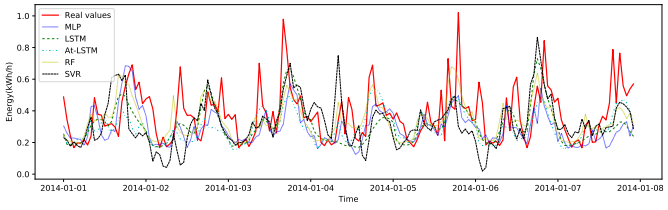
(a) **Scenario 1:** All buildings without influencing parameters.



(b) **Scenario 2:** All buildings with influencing parameters.



(c) **Scenario 3:** One private house without influencing parameters.



(d) **Scenario 4:** One private house with influencing parameters.

Fig. 9. Load forecast from the 1^{st} to the 7^{th} of January 2014.

The value of the two performance criteria (MAPE and RMSE) for the four scenario and the five implemented techniques are summarized in Table I.

TABLE I
COMPARISON OF THE IMPLEMENTED TECHNIQUES FOR THE FOUR SCENARIOS (BEST VALUE IN BOLD).

		deep MLP	deep LSTM	At-LSTM	SVR	RF
S1	RMSE	0.032	**0.030**	**0.030**	0.088	0.036
	MAPE	4.24	**3.90**	4.08	10.17	4.96
S2	RMSE	0.057	0.050	**0.047**	0.049	0.050
	MAPE	7.86	6.76	**6.12**	13.26	6.87
S3	RMSE	0.138	0.139	0.135	0.167	**0.134**
	MAPE	21.71	**20.9**	21.95	29.14	22.96
S4	RMSE	0.163	0.152	0.154	0.179	**0.137**
	MAPE	25.8	23.66	**23.21**	34.22	23.66

For hourly predictions, the consideration of influencing parameters during the training period do not seems to be

adaptive as the accuracy for scenarios 1 is higher (MAPE: $\sim$4 % vs $\sim$7 % respectively). This could be linked to the chosen influencing parameters or a de-synchronization in their time stamp. The size of the data-set or the chosen period (hour-ahead prediction) could be an explanation of this counter-intuitive result to be more deeply assessed.

We can say that Deep LSTM is a good technique for load prediction without influencing parameters as its MAPE is the smallest. Meanwhile, At-LSTM shows a better performance in case the influencing parameters are taken into account. RF provides comparable results to LSTM or At-LSTM and is better performing if you consider the RMSE values as the variation of energy consumption is important. The SVR is the worst technique in the four scenarios in this experience. Its MAPE and RMSE values are the highest and its inaccuracy (MAPE) is up to more than 34 % while the At-LSTM ones goes down to $\sim$23 %. For weekends, its prediction seems to be more accurate (not shown in this paper).

The computational tractability of the five implemented techniques is shown in Fig. 10.

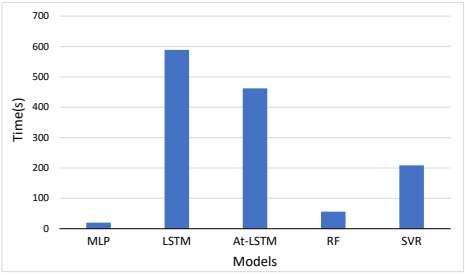


Fig. 10. Execution time of the five implemented techniques.

The kernel of SVR allows constituting a optimal hyperplane which aims to increase the accuracy of the prediction. In this issue, the training data-set is used with various parameters, the linear kernel can not give a better result than the nonlinear kernel. However, the chosen nonlinear kernel of the SVR makes its training time much longer than the linear one. Besides, MLP, and RF present moderate training times because they perform simple computations (e,g., the MLP model contains many units in which only linear operators are performed). Finally, LSTM presents the longest training time as it requires memory cells and more complex computations in each unit.

V. Conclusion

In this paper, we describe and implement five state of art machine learning models and applied them to short-term load forecasting to compare and analyze their performances on two relevant criteria. The idea is to help determine which one could be considered in the optimization loop of an energy management systems, either at the level of a district (local energy communities, or smart districts) or directly at the level of individual households (smart residential buildings). The five implemented techniques are deep multilayer perceptron (deep MLP), deep long short term memory (deep LSTM), attention-based LSTM, support vector regression (SVR), and random forest (RF), all of them chosen as the most efficient forecasting algorithm of the state of art for energy consumption prediction.

The comparison of the forecasting algorithms is presented with and without the consideration of influencing parameters (weather data for instance), based on real data (relying on a large training data-set). Two level of energy consumption are assessed: the single private household in which the variation of data is significant and the whole data-set, presenting a smart district or a local energy community, in which the variation of the data is much smoother. The prediction without influencing parameters seems to be more accurate in the considered scenarios, which is counter-intuitive at first, but interesting from the point of view of its application to energy management systems. The prediction on the single household data-set is promising, the mean absolute percentage error of the LSTM technique can be less than 4%. The RF, At-LSTM and deepLSTM are more efficient than the deep MLP or the SVR. To conclude, in the context of energy management systems, the At-LSTM is better fitting in cases where influencing parameters are taken into account. Otherwise, deep LSTM is the most interesting, if the long training time is not a problem.

The techniques compared in this paper represent the state of art of STLF in the energy context. To select one for an energy management system, sensitivity studies should be conducted on the training data-set duration as well as its impact on seasonal predictions. Also, a real-time approach could be enforced to improve accuracy in contexts where low-latency online predictions are needed.

References

[1] 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.

[2] E. A. Feinberg and D. Genethliou, "Load forecasting," in *Applied mathematics for restructured electric power systems*. Springer, 2005, pp. 269–285.

[3] E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Optimized parameter selection for assessing building energy efficiency," in *IEEE Young Researchers Symposium in Electrical Power Engineering (YRS)*, Ghent, Belgium, Apr. 2014.

[4] R. E. Edwards, J. New, and L. E. Parker, "Predicting future hourly residential electrical consumption: A machine learning case study," *Energy and Buildings*, vol. 49, pp. 591–603, 2012.

[5] J. Massana, C. Pous, L. Burgas, J. Melendez, and J. Colomer, "Short-term load forecasting in a non-residential building contrasting models and attributes," *Energy and Buildings*, vol. 92, pp. 322–330, 2015.

[6] K. Yun, R. Luck, P. J. Mago, and H. Cho, "Building hourly thermal load prediction using an indexed ARX model," *Energy and Buildings*, vol. 54, pp. 225–233, 2012.

[7] X. Wang, F. Fang, X. Zhang, Y. Liu, L. Wei, and Y. Shi, "LSTM-based short-term load forecasting for building electricity consumption," in *IEEE International Symposium on Industrial Electronics (ISIE)*, 2019, pp. 1418–1423.

[8] P.-H. Kuo and C.-J. Huang, "A high precision artificial neural networks model for short-term energy load forecasting," *Energies*, vol. 11, no. 1, p. 213, 2018.

[9] L. Sehovac and K. Grolinger, "Deep learning for load forecasting: Sequence to sequence recurrent neural networks with attention," *IEEE Access*, vol. 8, pp. 36 411–36 426, Feb. 2020.

[10] "Smart meters in London," Online, accessed: 2019-12-01. [Online]. Available: https://www.kaggle.com/jeanmidev/smart-meters-in-london/

[11] "Detail implementation of four scenarios," Online, accessed: 2020-20-04. [Online]. Available: https://colab.research.google.com/drive/