energies

MDPI

Article

Optimal Deep Learning LSTM Model for Electric Load Forecasting using Feature Selection and Genetic Algorithm: Comparison with Machine Learning Approaches †

Salah Bouktif [1,*], Ali Fiaz [1], Ali Ouni [2] and Mohamed Adel Serhani [1]

[1] Department of Computer Science and Software Engineering, College of Information Technology, UAE University, Al Ain 15551, UAE; alifiaz@uaeu.ac.ae (A.F.); Serhanim@uaeu.ac.ae (M.A.S.)
[2] Department of Software Engineering and IT, Ecole de Technologie Superieure, Montréal, QC H3C 1K3, Canada; ali.ouni@etsmtl.ca
* Correspondence: salahb@uaeu.ac.ae
† This work was supported by a UPAR grant from the United Arab Emirates University, under grant G00001930.

Received: 28 April 2018; Accepted: 21 May 2018; Published: 22 June 2018

Abstract: Background: With the development of smart grids, accurate electric load forecasting has become increasingly important as it can help power companies in better load scheduling and reduce excessive electricity production. However, developing and selecting accurate time series models is a challenging task as this requires training several different models for selecting the best amongst them along with substantial feature engineering to derive informative features and finding optimal time lags, a commonly used input features for time series models. Methods: Our approach uses machine learning and a long short-term memory (LSTM)-based neural network with various configurations to construct forecasting models for short to medium term aggregate load forecasting. The research solves above mentioned problems by training several linear and non-linear machine learning algorithms and picking the best as baseline, choosing best features using wrapper and embedded feature selection methods and finally using genetic algorithm (GA) to find optimal time lags and number of layers for LSTM model predictive performance optimization. Results: Using France metropolitan's electricity consumption data as a case study, obtained results show that LSTM based model has shown high accuracy then machine learning model that is optimized with hyperparameter tuning. Using the best features, optimal lags, layers and training various LSTM configurations further improved forecasting accuracy. Conclusions: A LSTM model using only optimally selected time lagged features captured all the characteristics of complex time series and showed decreased Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) for medium to long range forecasting for a wider metropolitan area.

Keywords: deep neural networks; long short term memory networks; short- and medium-term load forecasting; machine learning; feature selection; genetic algorithm

1. Introduction

Load forecasting enables utility providers to model and forecast power loads to preserve a balance between production and demand, to reduce production cost, to estimate realistic energy prices and to manage scheduling and future capacity planning. The primary criterion used for the classification of forecasting models is the forecasting horizon. Mocanu et al. [1] grouped electricity demand forecasting into three categories, short-term forecasts ranging between one hour and one

week, medium term ranging between one week and one year and long-term spanning a time of more than one year. The literature reveals that short-term demand forecasting has attracted substantial attention. Such forecasting is important for power system control, unit commitment, economic dispatch, and electricity markets. Conversely, medium and long-term forecasting was not sufficiently studied despite their crucial inputs for the power system planning and budget allocation [2].

In this paper, we focus on both short term and medium term monthly forecasting horizons. The rationale behind targeting simultaneously these two forecasting horizons is that deterministic models can be used successfully for both of them. However, in the case of long-term forecasting, stochastic models are needed to deal with uncertainties of forecasting parameters that always have a probability of occurrence [3].

Two challenges are associated with the targeted forecasting horizons. In the short-term case, the accuracy is crucial for optimal day-to-day operational efficiency of electrical power delivery and the medium term case, the prediction stability is needed for the precise scheduling of fuel supplies and timely maintenance operations. For prediction stability, a low forecast error should be preserved for the medium term span. Thus, the forecasting model should keep performing accurately or at least should not be excessively sensitive to the elapsed time within the medium term frame.

Although several electric load forecasting approaches using traditional statistical techniques, time series analysis and recent machine learning were proposed, the need for more accurate and stable load forecasting models is still crucial [4]. Recently, a particular attention is being paid to deep learning based approaches. These are based on artificial neural networks (ANNs) with deep architectures have gained attention of many research communities [5–7] due to their ability of capturing data behavior when considering complex non-linear patterns and large amounts of data. As opposed to shallow learning, deep learning usually refers to having a larger number of hidden layers. These hidden layers in deep network makes the model able to learn accurately complex input-output relations.

Long short-term memory (LSTM), a variation of deep Recurrent Neural Networks (RNN) originally developed Hochreiter et al. [8] to allow the preservation of the weights that are forward and back-propagated through layers. LSTM-based RNNs are an attractive choice for modeling sequential data like time series as they incorporate contextual information from past inputs. Especially, LSTM technique for time series forecasting has gained popularity due to its end-to-end modeling, learning complex non-linear patterns and automatic feature extraction abilities.

Time series models commonly use lags to make their predictions based on past observations. Selection of appropriate time lags for the time series forecasting is an important step to eliminate redundant features [9]. This helps to improve prediction model accuracy as well as gives a better understanding of the underlying process that generate the data. Genetic algorithm (GA), a heuristic search and optimization technique that mimics the process of evaluation with the objective to minimize or maximize some objective function [10] can be used to find appropriate number of lags for time series model. GA works by creating a population of potential answers to the problem to be solved, and then submit it to the process of evolution.

In this paper, we propose a LSTM-RNN-based model for aggregated demand side load forecast over short- and medium-term monthly horizons. Commonly used machine learning approaches are implemented to be compared to our proposed model. Important predictor variables and optimal lag and number of layers selection are implemented using feature selection and GA approaches. The performances of forecasting techniques are measured by using several evaluation metrics such as coefficient of variation RMSE (CVRMSE), mean absolute Error (MAE) and root mean square error (RMSE). Using France Metropolitan's electricity energy consumption data as a case study, our validation shows that LSTM-RNN based forecasting model outperforms the best of the studied alternative approaches with a high confidence.

The remainder of this paper is organized as follows: Section 2 provides an overview of the literature on load forecasting. Section 3 provides brief background on LSTM-RNN, benchmark machine learning models and on the forecasting evaluation metrics. Section 4 describes the methodology

of building our proposed forecast technique, its enhancement and validation. Section 5 describes experimental results and alternative time series validation approach. Section 6 provides discussion on threat to validity and Section 7 draws the conclusions.

2. Literature Review

Different models for load forecasting can be broadly classified as engineering methods and data driven methods. Engineering methods, also known as, physics based models; use thermodynamic rules to estimate and analyze energy consumption. Commonly, these models rely on context characteristics such as building structure, weather information, heating, ventilation, and air conditioning (HVAC) system information to calculate energy consumption. Physics based models are used in energy simulator software for buildings such as *EnergyPlus* and *eQuest*. The limitation of these models resides in their dependence on the availability and the accuracy of input information [11].

Data-driven methods, also known as artificial intelligence (AI) methods rely on historical data collected during previous energy consumption periods. These methods are very attractive for many groups of research, however little is known about their forecasting generalization. In other terms, their accuracy drops when they are applied on new energy data. The ability of generalization of data driven forecasting models remains an open problem. As reported for example in [12], the accuracy of forecasting results varies considerably for micro-grids with different capacities and load characteristics. Similar to load forecasting, data driven models for electricity price prediction were also proposed to help decision making for energy providing companies. Commonly used price forecasting techniques include multi-agent, reduced-form, statistical and computational intelligence as reviewed in [13].

Nowadays, the availability of relatively large amount of energy data makes it increasingly stimulating to use data-driven methods as an alternative to physics-based methods in load forecasting for different time horizons, namely, short, medium and long term [11]. Short-term load forecasting (STLF) has attracted more attention in the smart grid, microgrids and buildings and because of its usefulness for demand side management, energy consumption, energy storage operation, peak load anticipation, energy shortage risk reduction, etc. [14]. Many works were carried out for STLF. They ranged from classical time series analysis to recent machine learning approaches [15–20]. In particular, autoregressive and exponential smoothing models have remained the baseline models for time series prediction tasks for several years, however using these models necessitated careful selection of the lagged inputs parameter to identify the correct model configuration [21]. An adaptive autoregressive moving-average (ARMA) model developed by Chen et al. [22] to conduct day and week ahead load forecasts, reported superior performance compared to Box-Jenkins model. While these univariate time-series approaches directly model the temporal domain, they suffer from curse of dimensionality, accuracy dropping and require frequent retraining.

Various other approaches covering simple linear regression, multivariate linear regression, non-linear regression, ANN and support vector machines (SVMs) were applied for electricity demand forecasting primary for the short-medium term [4,23,24] and lesser for the long term [4]. Similarly, for price forecasting, Cincotti et al. used [25] proposed three univariate models namely, ARMA-GARCH, multi-layer perceptron and support vector machines to particularly predict day ahead electricity spot prices. Results showed that SVMs performed much better than the benchmark model based on the random walk hypothesis but close to the ARMA-GARCH econometric technique. For a similar forecasting aim, a combination of wavelet transform neural network (WTNN) and evolutionary algorithms (EA) was implemented for day-ahead price forecasting. This combination resulted in more accurate and robust forecasting model [26].

Noticeably, ANN was widely used for various energy-forecasting tasks because of its ability to model nonlinearity [4]. Hippert, et al., in their review work [20], have summarized most of the applications of ANN to STLF as well as its evolution. Furthermore, various structures of ANN have found their ways into load forecasting in order to improve models accuracy. Namely, the most common configuration of ANN, multilayer perceptron (MLP), was used to forecast a load profile using previous

load data [27], the fuzzy neural [28], wavelet neural networks [29], fuzzy wavelet neural network [30] and self-originating map (SOM), neural network [31], were used mainly for STLF.

Unlike the conventional ANN structures, a deep neural network (DNN) is an ANN with more than one hidden layer. The multiple computation layers structure increases the feature abstraction capability of the network, which makes them more efficient in learning complex non-linear patterns [32]. To the best of our knowledge and according to Ryu et al. [14] in 2017, only a few DNN-based load forecasting models are proposed. Recently, in 2016, Mocanu et al. [1] employed, a deep learning approach based on a restricted Boltzmann machine, for single-meter residential load forecasting. Improved performance was reported compared to shallow ANN and support vector machine. The same year (2016), Marino et al. [33] presented a novel energy load forecasting methodology using two different deep architectures namely, a standard LSTM and an LSTM with sequence to Sequence architecture that produced promising results. In 2017, Ryu et al. [14] proposed a DNN based framework for day-ahead load forecast by training DNNs in two different ways, using a pre-training restricted Boltzmann machine and using the rectified linear unit without pre-training. This model exhibited accurate and robust predictions compared to shallow neural network and others alternatives (i.e., double seasonal Holt-Winters model and the autoregressive integrated moving average model). In 2018, Rahman et al. [34] proposed two RNNs for medium to long-term electric forecast for residential and commercial buildings. A first model feeds a linear combination of vectors to the hidden MLP layer while a second model applies a shared MLP layer across each time step. The results were more accurate than using a 3-layer multi-layered perceptron model.

Seemingly closer to our current work, but technically distinct from it, Zheng et al. [35] developed a hybrid algorithm for STLF that combined similar days selection, empirical mode decomposition and LSTM neural networks. In this work, Xgboost was used to determine features importance and k-means was employed to merge similar days into one cluster. The approach substantially improved LSTM predictive accuracy.

Despite the success of machine learning models, in particular the late deep learning, to perform better than traditional time series analysis and regression approaches, there is still a crucial need for improvement to better model non-linear energy consumption patterns while targeting high accuracy and prediction stability for the medium term monthly forecasting. Definitely, we share the opinion that optimally trained deep learning model derives more accurate forecasting outputs than those obtained with a shallow structure [36]. Furthermore, we believe that the adoption of deep learning in solving load-forecasting problem needs more maturity though increasing the number of works with different forecasting configurations. These configurations include the aim and horizon of forecasting, the type or nature of inputs, the methods of determining their relevance, the selected variation of deep learning and the way of validating the derived forecasting model. For instance, using lagged features as inputs to enable deep models to see enough past values relevant for future prediction can cause overfitting if too many lags are selected. This is because deep models with lot of parameters can overfit due to increase in dimensionality. In this paper, we aim at proving that optimal LSTM-RNN will behave similarly in the context of electric load forecasting for both the short-and-medium horizon. Accordingly, our approach differs from the previous deep learning models in that:

(i) Implementing feature importance using wrapper and hybrid methods, optimal lag and number of layers selection for LSTM model using GA enabled us to prevent overfitting and resulted in more accurate and stable forecasting.
(ii) We train a robust LSTM-RNN model to forecast aggregate electric load for short- and medium term horizon using a large dataset for a complete metropolitan region covering a period of nine years at a 30 min resolution
(iii) We compare the LSTM-RNN model with the machine learning benchmark that is performing the best among several linear and non-linear models optimized with hyperparameter tuning.

3. Background

This section provides brief backgrounds on LSTM-RNN, on benchmark machine learning models and on the forecasting evaluation metrics.

3.1. From RNN to LSTM-RNN's

A RNN is a special type of ANN that makes use of sequential information due to directed connections between units of an individual layer. They are called recurrent because they perform the same task for every element in the sequence. RNN are able to store memory since their current output is dependent on the previous computations. However, RNNs are known to go back only a few time steps due to vanishing gradient problem. Rolled and unrolled RNN configuration over the input sequence is shown in Figure 1 (adopted from ([37])).

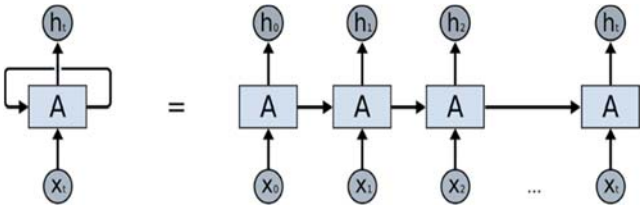

Figure 1. Recurrent neural networks (RNN) and the unfolding in time of the computation.

Since Standard RNNs suffer from vanishing and exploding gradient problems, LSTMs were specially designed to overcome these problems by introducing new gates which allow a better control over the gradient flow and enable better preservation of long-range dependencies. The critical component of the LSTM is the memory cell and the gates shown in Figure 2 (adopted from ([37])).

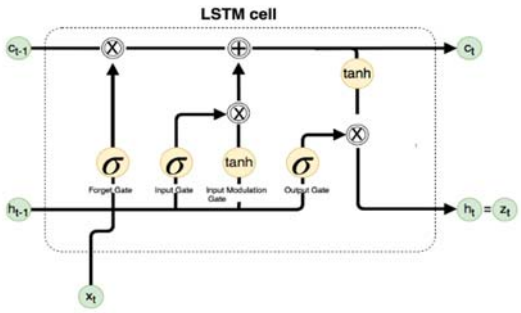

Figure 2. Information flow in a long short-term memory (LSTM) block of the RNN.

These gates in LSTM cell enables it to preserve a more constant error that can be back propagated through time and layers allowing recurrent nets to continue to learn over many time steps [38]. These gates work in tandem to learn and store long and short-term sequence related information. The RNN models its input sequence $\{x_1, x_2, ..., x_n\}$ using the recurrence:

$$h_t = f(h_{t-1},\ x_t) \tag{1}$$

where x_t is the input at time t, and h_t is the hidden state. Gates are introduced into the recurrence function f in order to solve the gradient vanishing or explosion problem. States of LSTM cells are computed as follows:

$$i_t = \sigma(W_i \cdot [h_{t-1},\ x_t] + b_i) \tag{2}$$

$$f_t = \sigma\left(W_f \cdot [h_{t-1},\ x_t] + b_f\right) \tag{3}$$

$$o_t = \sigma(W_o \cdot [h_{t-1},\ x_t] + b_o) \tag{4}$$

$$\widetilde{C} = tanh(W_C \cdot [h_{t-1},\ x_t] + b_C) \tag{5}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C}_t \tag{6}$$

$$h_t = o_t \odot \tanh(C_t) \tag{7}$$

In the equations above, i_t, f_t and o_t are the input, forget and output gates respectively. W's and b's are parameters of the LSTM unit, C_t is the current cell state and $\widetilde{C}$ is new candidate values for cell state. There are three sigmoid functions for i_t, f_t and o_t gates that modulates the output between 0 and 1 as given in Equations (2)–(4). The decisions for these three gates are dependent on the current input x_t and the previous output h_{t-1}. If the gate is 0, then the signal is blocked by the gate. Forget Gate f_t defines how much of the previous state h_{t-1} is allowed to pass. Input gate i_t decides which new information from the input to update or add to the cell state. Output gate o_t decides which information to output based on the cell state. These gates work in tandem to learn and store long and short-term sequence related information.

The memory cell C acts as an accumulator of the state information. Update of old cell state C_{t-1} into the new cell state C_t is performed using Equation (6). Calculation of new candidate values $\widetilde{C}$ of memory cell and output of current LSTM block h_t uses hyperbolic tangent function as in Equations (5) and (7). The two states cell state and the hidden state are being transferred to the next cell for every time step. This process then continues to repeat. Weights and biases are learnt by the model by minimizing the differences between the LSTM outputs and the actual training samples.

3.2. Alternative Modeling Approaches

Data-driven approaches have addressed variety of energy prediction and load forecasting tasks and thus has attracted significant research attention [39]. Common data driven approaches for time series data includes linear regression that fits the best straight line through the training data and using ordinary least square method to estimate the parameters by minimizing the sum of the squared vertical distances. Ridge regression is another linear model that addresses the issue of multi-collinearity by penalizing the extreme values of the weight vector. These linear regression models are relatively easier to develop and interpret. K-nearest neighbor is a non-parametric model where the prediction is the average value of the k-nearest neighbors. This algorithm is intuitive, easy to implement and can give reliable results for electricity load forecasting when its parameters are properly tuned. Ensemble models like random forest and extra trees use ensemble of decision trees to predict the output and thus avoid the problem of overfitting encountered by single decision tree. These models have low sensitivity to parameter values and can produce accurate forecasts. Gradient boosting is another ensemble model that uses ensemble of weak learners in order to help increase accuracies of trees by giving more weight to wrong predictions.

3.3. Performance Metrics for Evaluation

Commonly used metrics to evaluate forecast accuracy are the coefficient of variation (CV RMSE), the root mean squared error (RMSE) and the MAE [40]. CV (RMSE) is the RMSE normalized by the mean of the measured values and quantifies typical size of the error relative to the mean of the observations. A high CV score indicates that a model has a high error range. MAE, a commonly used metric, is the mean value of the sum of absolute differences between actual and forecasted. RMSE is another commonly used metric. It penalizes the larger error terms and tends to become increasingly larger than MAE for outliers. The error measures are defined as follows:

$$CV(RMSE)\% = \sqrt{\frac{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\overline{y}}} \tag{8}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}} \quad (9)$$

$$MAE = \sqrt{\frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{n}} \quad (10)$$

where $\hat{y}$ is the predicted, y_i is the actual and $\overline{y}$ is the average energy consumption.

4. Methodology

In this section, we describe the proposed methodology process for short and medium-term load forecasting LSTM-RNNs as depicted in Figure 3. The proposed methodology process can be seen as a framework of four processing components, namely, data preparation and preprocessing component, the machine learning benchmark model construction component, LSTM-RNN training component and LSTM-RNN validation component. In the following, we present an overview of the methodology, and then we describe for each methodology components its detailed mission followed by an illustration on the *Réseau de Transport d'Électricité* (RTE) power consumption data set, our case study.

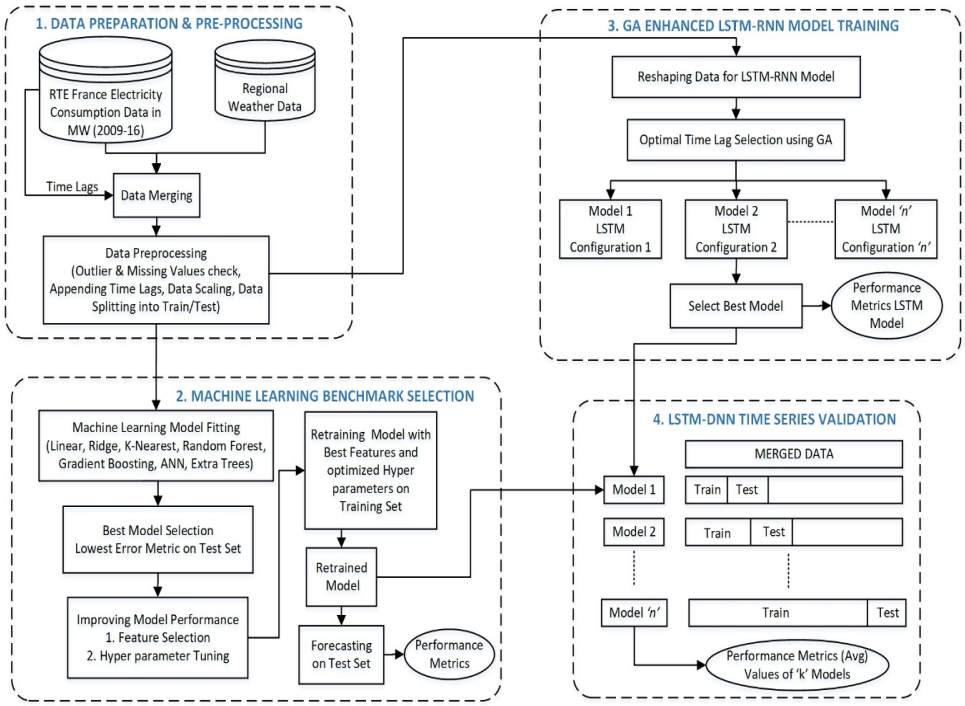

Figure 3. Proposed forecasting methodology.

4.1. Methodology Process Overview

The first processing step starts by merging the electric energy consumption data with weather data and time lags. The merging is performed because weather data and time lags are known to influence power demand. Then a preprocessing of data is carried out in order to check null values and outliers, scale the data to a given range and split the time series data into train and test subsets while retaining the temporal order. This step aims at preparing and cleaning data to be ready for the further analysis.

In the second processing step of our framework, benchmark model will be selected by fitting seven different linear and non-linear machine-learning algorithms to the data and choosing the model that performs the best. Further improvement in accuracies of selected model will be achieved by performing feature engineering and hyerparameter tuning. This benchmark would then be used for forecasting

and its obtained performance would be compared with that of the LSTM-RNN model. In the third processing step of the process, a number of LSTM models with different model configurations like number of layers, number of neurons in each layer, training epochs, optimizer etc. will be tested. Optimal number of time lags and LSTM layers would be selected using GA. The best performing configuration will be determined empirically after several trails with different settings, and then it would be used for the final LSTM-RNN model.

4.2. Data Preparation and Pre-Processing

Time series forecasting is a special case of sequence modeling. The observed values correlate with their own past values, thus individual observations cannot be considered independent of each other. Exploratory analysis of electric load time-series can be useful to identifying trends, patterns and anomalies. We will plot electric load consumption box plot for various years, quarters and weekday indicator, which would give us a graphical display of data dispersion. Furthermore, correlation of electric consumption with time lags and weather variables will also be investigated to check for the strength of association between these variables.

4.2.1. Preliminary Data Analysis

Our research makes use of a RTE power consumption data set [41], which gives us a unique opportunity to predict next half-hourly electrical consumption in MW containing nine years' data in metropolitan France. The power consumption dataset ranges from January 2008 until December 2016. The load profile from January to February 2011 as depicted in the Figure 4 follows cyclic and seasonal patterns, which can be related to human, industrial and commercial activities.

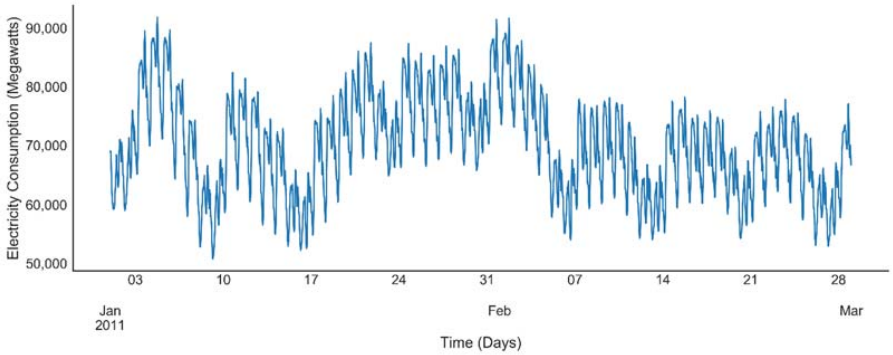

Figure 4. Electricity load versus time (January–February 2011).

Box plots of load consumptions reveal that average load is almost constant across years (Figure 5a) while quarterly plot shows low consumption in second and third quarter as compared to others (Figure 5b). Power demand in France increases in first and fourth quarters due to heating load in winter months.

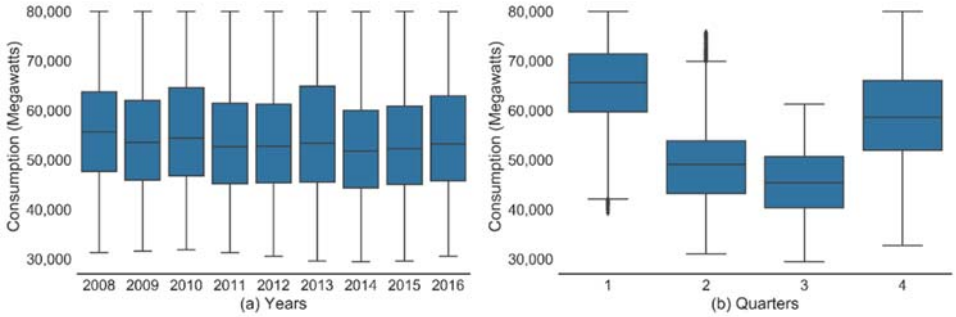

Figure 5. Box Plot of Electric load (**a**) Yearly (**b**) Quarterly.

Holidays and weekends can affect the usage of electricity, as the usage patterns are generally quite different from usual. This can be used as a potential feature to our model. Figure 6 shows the box plot of consumption for weekdays and weekend. The weekend consumption is high compared to weekdays across all years.

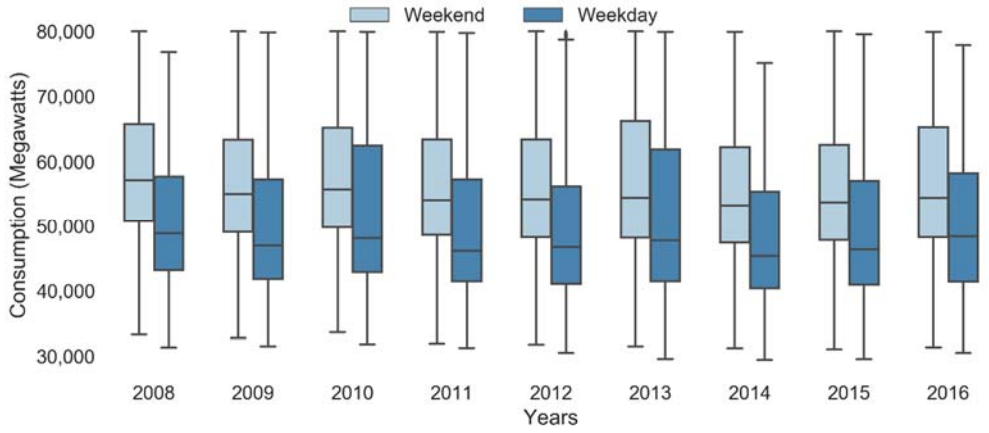

Figure 6. Box Plot of Electric Load Consumption Weekend vs. Weekday.

Correlation matrix of electric consumption showed a high correlation with previous several time lags (0.90–0.98). Amongst the weather variables, temperature had a high negative correlation with consumption since we are analyzing a cooling load. Humidity and wind speed have low correlation. Figure 7 shows scatter plot of weather variables with consumption.

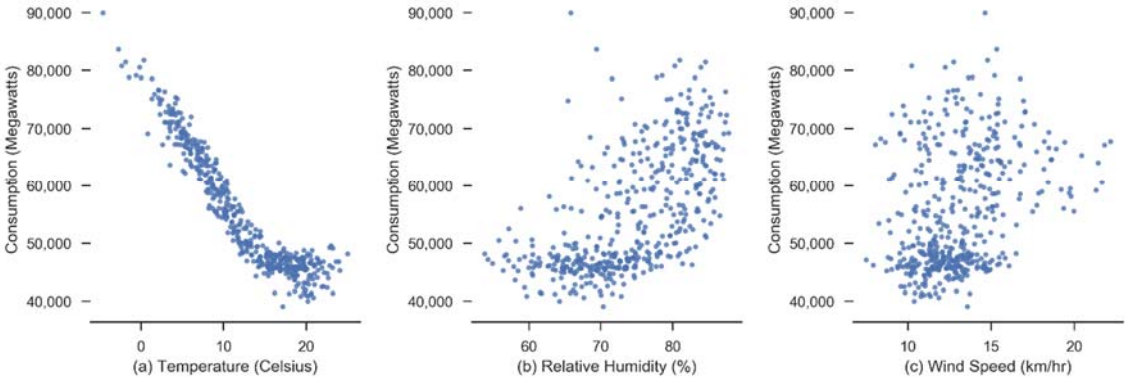

Figure 7. Scatter Plot of Electric load vs. (**a**) Temperature (**b**) Humidity (**c**) Wind Speed.

4.2.2. Data Pre-Processing

Data preprocessing is a vital step to obtain better performance and accuracies of machine learning as well as deep learning-based models. It is about dealing with inconsistent, missing and noisy data. Our dataset comes from RTE, which is the electricity transmission system operator of France. It has measurements of electric power consumption in metropolitan France with a thirty-minute sampling rate. There are two categories of power in the dataset, namely, definitive and intermediate. Here we will only use definitive data. Weather data comprising outdoor temperature, humidity and wind speed are merged as exogenous input with the consumption data. Further data preprocessing comprised data cleansing, normalization, and structure change. As machine learning and LSTM-RNN models are sensitive to the scale of the inputs, the data are normalized in the range [0, 1] by using feature scaling.

The data is split into train and test set while maintaining the temporal order of observations. Test data is used for evaluating accuracy of the proposed forecasting model and not used in training

step. Standard practice for splitting data is performed using 80/20 or 70/30 ratios for machine learning models [42]. The last 30 percent of the dataset is withheld for validation while the model is trained on the remaining 70 percent of the data. Since our data is large enough, both training and tests sets are highly representative of the original problem of load forecasting. Stationarity of time series is a desired characteristic to use Ordinary Least Square regression models for estimation of regression relationship. Regressing non-stationary time series can lead to spurious regressions [43]. High R^2 and high residual autocorrelation can be signs of spurious regression. Dickey-Fuller test is conducted to check stationarity. The resulting *p*-value is less 0.05, thus we reject the null hypothesis and conclude that the time series is stationary.

4.3. Selecting the Machine Learning Benchmark Model

Benchmarking is an approach that demonstrate new methodologies abilities to run as expected and thus comparing the result to existing methods [44]. We will use linear regression, ridge, regression k-nearest neighbors, random forest, gradient boosting, ANN and extra trees Regressor as our selected machine learning models. The initial input to these models is the complete set of features comprising time lags, weather variables temperature, humidity, wind speed and schedule-related variables, month number (between 1 and 12), quarter and weekend or weekday. Use lagged versions of the variables in the regression model allows varying amounts of recent history to be brought into the forecasting model. Selected main parameters for various machine-learning techniques are shown in Table 1.

Table 1. Parameters for Machine Learning Techniques.

No.	Model	Parameters
1	Ridge Regression	Regularization parameter, α = 0.8
2	k-Nearest Neighbor	No. of neighbors, n = 5, weight function = uniform, Distance Metric = Euclidian
3	Random Forest	No of Trees = 125, max depth of the tree = 100, min samples split = 4, min sample leaf = 4
4	Gradient Boosting	No of estimators = 125, maximum depth = 75, min samples split = 4, min sample leaf = 4
5	Neural network	Activation = relu, weight optimization = adam, batch size = 150, number of epochs = 300, learning rate = 0.005
6	Extra Trees	No of Trees = 125, max depth of the tree = 100, min samples split = 4, min sample leaf = 4

All the implemented machine-learning models utilized in this study used the mean squared error as the loss function to be minimized. All trained models are evaluated on the same test set and performance is measured in terms of our evaluation metrics. For our baseline models comparison, we will leave parameters to their default values. Optimal parameters for the best model will be chosen later. The results are shown in Table 2.

Table 2. Performance Metrics of Machine Learning Models.

Model	RMSE	CV (RMSE)	MAE
Linear Regression	847.62	1.55	630.76
Ridge	877.35	1.60	655.70
k-Nearest Neighbor	1655.70	3.02	1239.35
Random Forest	539.08	0.98	370.09
Gradient Boosting	1021.55	1.86	746.24
Neural network	2741.91	5.01	2180.89
Extra Trees	466.88	0.85	322.04

ANN models are known to overfit the data and show poor generalization when the dataset is large and training times are longer [45]. From the above results, it is seen that ANN does not perform well. This is because in order to achieve good accuracies, ANN model needs to have optimal network structure and update weights, which may require several epochs and network configurations. During training of our models, it was also noticed that ANN took the longer time as compared to other approaches, which is an indication that the model overfits and as the result shows poor test set performance.

Since the ensemble approach of Extra Trees Regressor model is giving the best results, therefore we will use this as our benchmark model. The main difference of Extra Trees Regressor with other tree based ensemble methods is randomization of both attribute and cut-point choice while splitting a tree node. Since this model performs the best amongst all other models, it is selected as our benchmark.

4.3.1. Improving Benchmark Performance with Feature Selection & Hyper Parameter Tuning

Wrapper and embedded are model based feature selection methods that can help remove redundant features and thus obtain better performing and less overfitting model. In wrapper method, feature importance is assessed using a learning algorithm while in embedded methods the learning algorithm performs feature selection as well in which feature selection and parameter selection space are searched simultaneously. We will use regression and ensembles based methods to identify both linear and non-linear relationships among features and thus ascertain both relevant and redundant features. In the regression case, recursive feature elimination works by creating predictive models, weighting features, and pruning those with the smallest weights. In the ensemble case, the Extra Trees Regressor decides feature importance based on the decrease in average impurity computed from all decision trees in the forest without making any assumption whether our data is linearly separable or not. Both the regression and ensemble-based models emphasized time lags as the most important features in comparison to weather and schedule related variables. With respect to our data set, among the weather and schedule related features, temperature and quarter number were found to be important after time lags. Figure 8 shows the relative importance of each feature. Subsequently, the input vector to machine learning models should include time lags as features.

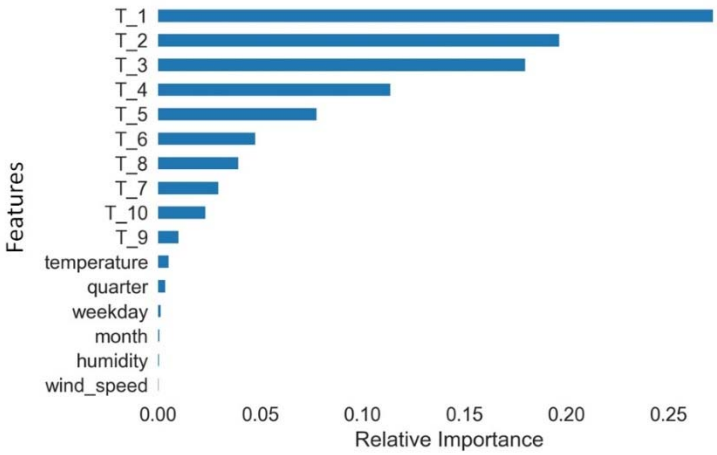

Figure 8. Feature importance plot.

Various combinations of 5, 10, 20, 30 and 40 time lags were evaluated for Extra Trees Regressor model by training several models. Result showed using only 30 lagged variables as features or combination of lagged features with temperature and quarter produced similar performance for this model. This is an indication that temporal load brings information that is originally contained within the weather parameters.

Grid Search cross validation [46] is a tuning method that uses cross validation to perform an exhaustive search over specified parameter values for an estimator. This validation method is used for hyperparameters tuning of our best Extra Trees Regressor model by using a fit and score methodology to find the best parameters. After tuning the best model, Extra Trees Regressor model is found to have 150 estimators with max depth of 200. Table 3 shows the performance results of the best Extra Trees Regressor obtained after tuning.

Table 3. Performance metrics of best model after feature selection & hyperparameter tuning.

Metrics	Values After	Values Before
RMSE	428.01	466.88
CV(RMSE) %	0.78	0.85
MAE	292.49	322.04

4.3.2. Checking Overfitting for Machine Learning Model

With too many features, the model may fit the training set very well but fail to generalize to new examples. It is necessary to plot learning curve that shows the model performance on training and testing data over a varying number of training instances.

Mean squared errors for both the training and testing sets for our optimized Extra Tree model (i.e., after hyperparameter optimization) decrease with bigger sizes of training data and converge at similar values, which shows that our model is not overfitting, as reported in Figure 9.

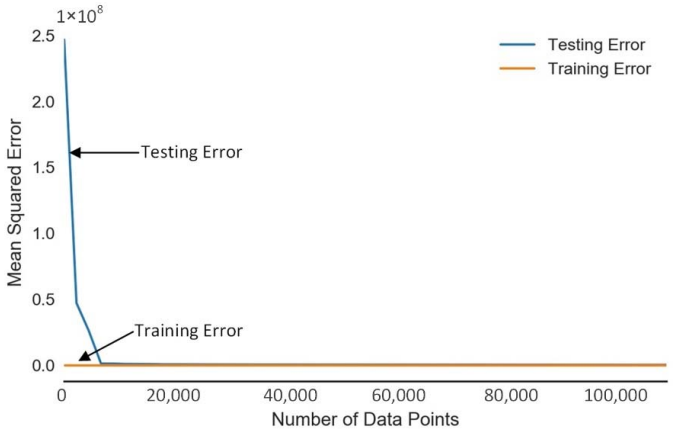

Figure 9. Learning curve for extra trees regressor.

4.4. GA-Enhanced LSTM-RNN Model

As seen from the machine learning modeling results in the previous section, there was almost no change in the model performance when weather and schedule related variables were removed and only the time lags were used as inputs. Therefore, based on the results of feature importance and cyclical patterns identified in the consumption data, we will only use time lags for LSTM model to predict the future. These time lags are able to capture conditional dependencies between successive time periods in the model. A number of different time lags were used as features to machine learning model, thereby training model several times and selecting the best lag length window. However, experimenting the same with LSTM deep network which has large number of parameters to learn would be computationally very expensive and time consuming.

Finding optimal number of lags and number of hidden layers for LSTM model is a non-deterministic polynomial (NP) hard problem. Meta-heuristic algorithms like GA do not guarantee to find the global optimum solution, however, they do tend to produce suboptimal good solutions that are sometimes near global optimal. Selecting optimal among a large number of potential combinations

is thus a search and optimization problem. Previous research has shown that GA algorithms can be effectively used to find a near-optimal set of time lags [47,48]. GA provides solution to this problem by an evolutionary process inspired by the mechanisms of natural selection and genetic science. The pseudo-code for the GA algorithm, crossover and mutation operations are shown in Figure 10.

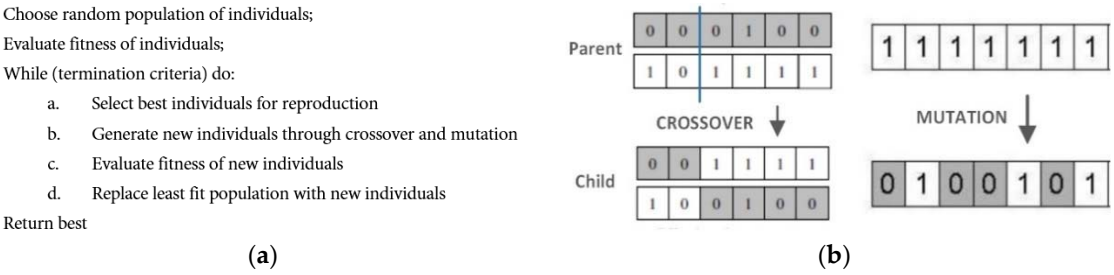

Figure 10. (**a**) Pseudo code for genetic algorithm (GA); (**b**) crossover and mutation operations.

4.4.1. GA Customization for Optimal Lag Selection

Choosing an optimal number of time lags and LSTM layers to include is domain specific and it could be different for each input feature and data characteristics. Using many lagged values increases the dimensionality of the problem, which can cause the deep model to overfit as well as difficult to train. GA inspired by the process of natural selection is used here for finding optimal number of time lags and layers for LSTM based deep network. Number of time lags will be tested from 1 to 99 and number of hidden layers from 3 to 10. A binary array randomly initialized using Bernoulli distribution is defined as a genetic representation of our solution. Thus, every chromosome represents an array of time lags. Population size and number of generations is set to 40. Various operators of GA are set as follows:

(i) *Selection:* roulette wheel selection was used to select parents according to their fitness. Better chromosomes have more chances to be selected

(ii) *Crossover:* crossover operator exchanges variables between two parents. We have used two-point crossover on the input sequence individuals with crossover probability of 0.7.

(iii) *Mutation:* This operation introduce diversity into the solution pool by means of randomly swapping bits. Mutation is a binary flip with a probability of 0.1

(iv) *Fitness Function:* RMSE on validation set will act as a fitness function.

GA solution would be decoded to get integer time lags size and number of layers, which would then be used to train LSTM model and calculate RMSE on validation set. The solution with highest fitness score is selected as the best solution. We reduced the training and test set size in order to speed up the GA search process on a single machine. The LSTM-RNN performed poorly on smaller number of time lags. Its most effective length found was 34 time lags with six hidden layers as shown in Figure 11.

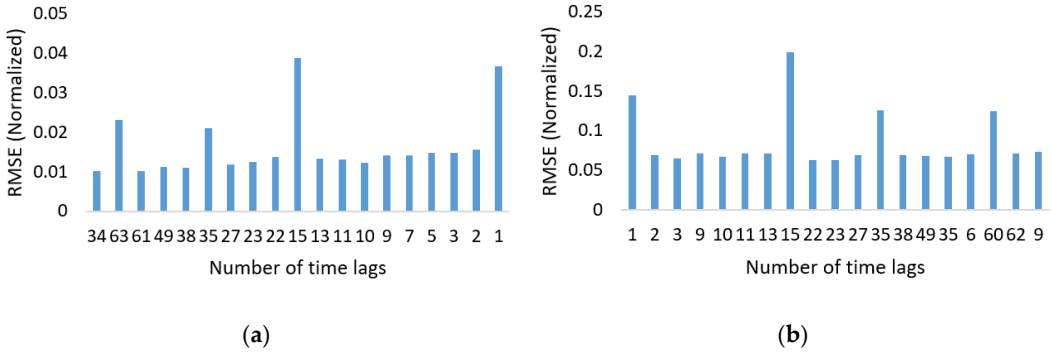

Figure 11. Root mean square error (RMSE) vs. time lags for (**a**) six and (**b**) four hidden layers.

4.4.2. GA-LSTM Training

To model our current forecasting problem as a regression one, we consider univariate electric load time series with N observations $\{X_{t_1}, X_{t_2}, \ldots, X_{t_N}\}$. The task of forecasting is to utilize these N data points to predict the next H data points $\{X_{t_{N+1}}, X_{t_{N+2}}, \ldots, X_{t_{N+H}}\}$ in the future of the existing time series. LSTMs are specifically designed for sequential data that exhibits patterns over many time steps.

Generally the larger the dataset, more hidden layers and neurons can be used for modeling without overfitting. Since our dataset is large enough, we can achieve higher accuracy. If there are too less neuron per layer, our model will under fit during training. In order to achieve good performance for our model we aim to test various model parameters including number of hidden layers, number of neurons in each layer, number of epochs, batch sizes, activation and optimization function.

Neurons in the input layer of LSTM model matches the number of time lags in the input vector, hidden layers are dense fully connected and output layer has a single neuron for prediction with linear activation function. Means squared error is used as the loss function between the input and the corresponding neurons in the output layer.

5. Experimental Results: GA-LSTM Settings

This section is dedicated to empirically set the parameters of LSTM, to validate the obtained model and to discuss the summary results. After finding optimal lag length and number of layers, various other meta-parameters for the LSTM-RNN model were tested as follows:

1. The number of neurons in the hidden layers were tested from 20 to 100.
2. size was varied from 10 to 200 training examples and training epochs from 50 to 300.
3. Sigmoid, hyperbolic tangent (tanh) and rectified linear unit (ReLU) were tested as the activation functions in hidden layers.
4. gradient descent (SGD), Root Mean Square Propagation (RMSProp) and adaptive moment estimation (ADAM) were tested as optimizers.

We got significantly better results by having six hidden layers having 100, 60 and 50 neurons. The number of epochs used were 150 with batch size of 125 training examples. For our problem, nonlinear activation function ReLu performed the best and is thus used as activation function for each of the hidden layers. ReLU does not encounter vanishing gradient problem as with tanh and sigmoid activations. Amongst optimizers, ADAM performed the best and showed faster convergence than the conventional SGD. Furthermore, by using this optimizer, we do not need to specify and tune a learning rate as with SGD.

The final selected parameters were the best result available from within the tested solutions. Due to the space limitation, we cannot provide all the test results for different network parameters. Table 4 shows the summary results of comparing LSTM-RNN with the Extra Trees Regressor model.

Table 4. Performance metrics of the long short-term memory (LSTM) Model on the test set.

Metrics	LSTM Metrics 30 Lags	LSTM Metrics Optimal Time Lags	Extra Tree Model Metrics	Error Reduction (%)
RMSE	353.38	341.40	428.01	20.3
CV(RMSE)	0.643	0.622	0.78	20.3
MAE	263.14	249.53	292.49	14.9

Plot of actual data against the predicted value by the LSTM model for the next two weeks, as medium term prediction. Figure 12 shows a good fit and stable prediction for the medium term horizon.

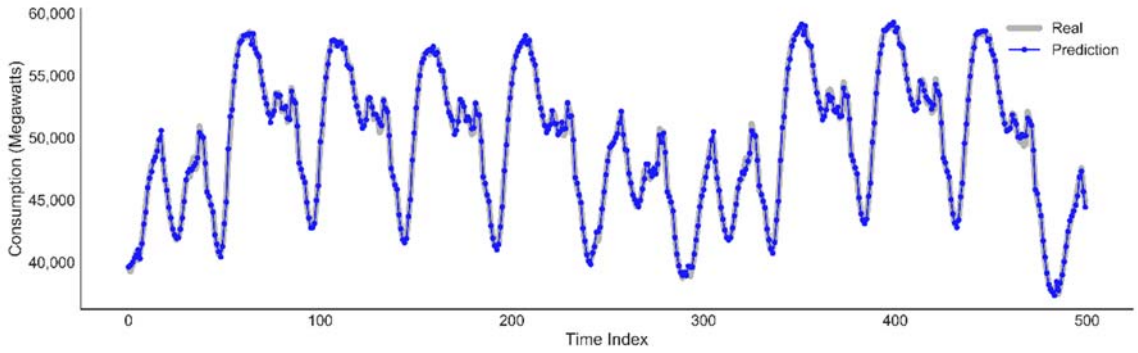

Figure 12. Actual vs. predicted forecast by the LSTM model.

5.1. Cross Validation of the LSTM Model

Time series split is a validation methods inspired by k-fold cross validation suitable for time series case [49]. The method involves repeating the process of splitting the time series into train and test sets 'k' times. The test size remains fixed while training set size will increase for every fold as in Figure 13. The sequential order of time series will be maintained. This additional computation will offer a more robust performance estimate of both our models on sliding test data.

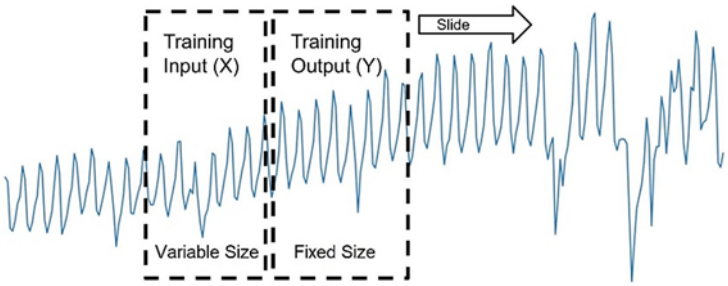

Figure 13. Time Series Split for Model Validation.

Indeed, we use 5-fold cross validation to train five Extra Trees and five LSTM-RNN models with fixed test size and increasing training set size in every fold, while maintaining the temporal order of data. Mean performance metrics of the results are plotted in Figure 14.

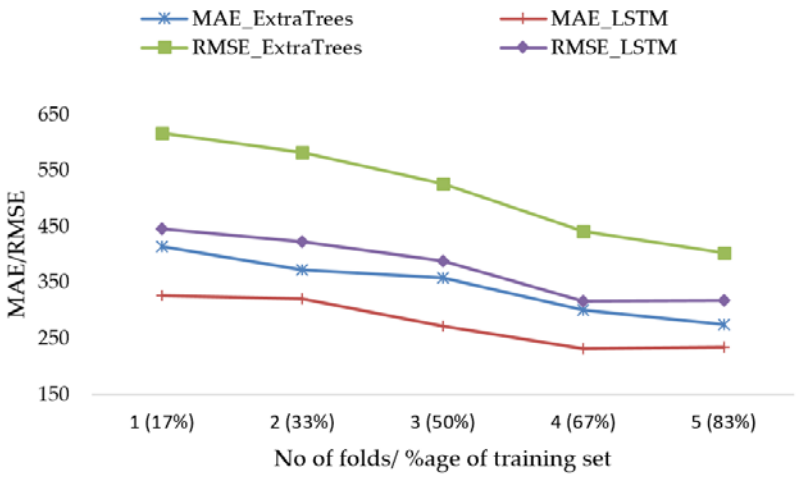

Figure 14. Time series cross validation with Multiple Train-Test Splits.

Mean and standard deviation values are collected on RMSE, CV (RMSE) and MAE for LSTM-RNN model as well as for the best Extra Trees Regressor model. These results are shown in Table 4. They confirm our previous results in Table 5. Furthermore, LSTM showed lower forecast error compared to Extra Trees Model using this validation approach.

Table 5. Performance Metrics of LSTM-RNN and Extra Tree Regressor Model using the Cross Validation Approach.

Model	Mean	Std. Deviation
RMSE Extra Trees	513.8	90.9
RMSE LSTM	378	59.8
CV (RMSE) % Extra Trees	1.95	0.3
CV (RMSE) % LSTM	1.31	0.2
MAE Extra Trees	344	55.8
MAE LSTM	270.4	45.4

5.2. Short and Medium Term Forecasting Results

In order to check the model performance on short (i.e., accuracy) and medium term (i.e., stability) horizons, random sample of 100 data points were selected uniformly from various time ranges from one week to several months as in Table 6. The means of the performance metrics values is computed of each range. Results show that error values are consistent with low standard deviation. The accuracy measured by CV (RMSE) is 0.61% for the short term and an average of 0.56% for the medium term for the medium term. With very comparable performances for both the short term and the medium term horizons, it is obvious that our LSTM-RNN is accurate and stable.

Table 6. Performance Metrics of LSTM-RNN on various Time Horizons.

Forecasting Horizon	MAE	RMSE	CV (RMSE) %
2 Weeks	251	339	0.61
Between 2–4 Weeks	214	258	0.56
Between 2–3 Months	225	294	0.63
Between 3–4 Months	208	275	0.50
Mean-Medium term	215.6	275.6	0.56
Std. Dev.	8.6	18	0.06

6. Discussion: Threat to Validity

In this section, we respond to some threats to construct validity, internal validity and external validity of our study. The construct validity in our study is concerned with two threats. The first one is related to sufficiency of time series data to well predict the energy load. This question was taken into account from the beginning of the methodology process and other source of data; in particular, weather information was added as the input rather than excluded by different feature selection techniques. The second threat is concerned with the algorithms used in the different steps and components of the methodology that could use some unrealistic assumptions inappropriate for our case study context. To avoid such a threat we have developed all the algorithms ourselves.

The internal validity in our case is concerned with three main threats. The first one is related to how much the data set is suitable to train a LSTM-RNN model that needs a large amount of data to mine historical patterns. The RTE power consumption data set ranges from January 2008 until December 2016. This data set is large enough to allow deep learning that is particularly targeted by LSTM-RNN. The second threat is LSTM-RNN overfitting. It can be observed by the learning curve that describes error rate according to the number of epochs. When overfitting occurs, after a certain point test error starts increasing while training error still decreases. Too much training with complex forecasting model leads to bad generalization. To prevent overfitting for our LSTM-RNN model,

we have randomly shuffled the training examples at each epoch. This helps to improve generalization. Furthermore, early stopping is also used whereby training is stopped at the lowest error achieved on the test set. This approach is highly effective in reducing overfitting. The diagnostic plot of our LSTM-RNN model in Figure 15 shows that the train and validation losses decrease as the number of epochs increases. The loss stabilizes around the same point showing a good fit. The third threat is related to the choice of the alternative machine learning for comparison. We may "miss-represent" the spectrum of techniques used for the energy prediction problem. For this concern, we have selected the most successful techniques according to our literature exploration. A stronger strategy to avoid this threat is to carry out a specific study to identify the top techniques for each class of prediction circumstances. This latter is one of the objectives of our future works.

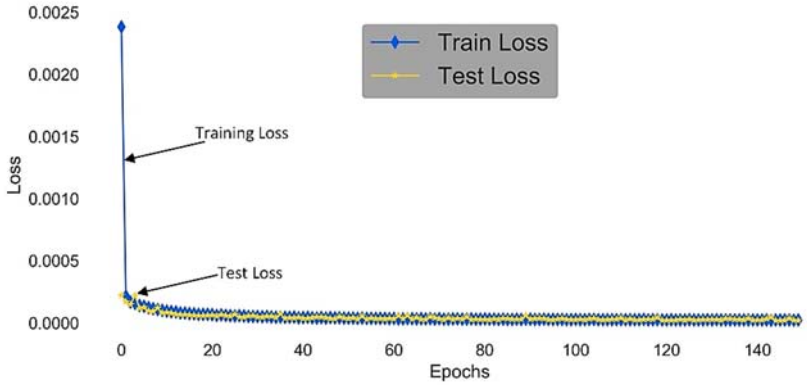

Figure 15. LSTM learning curve.

The external validity in our case is concerned with the generalization of our results. In other terms, the LSTM-RNN may not perform well when it is used in very different circumstances of energy consumption. This threat is mitigated by two choices. The first is the fact that we are using a deep learning approach that is known to recognize good patterns and from a large amount of data and from the far history in the case of times series. Deep learning is a variation of ANN that produce a robust model. The second choice is the use of a large amount of training data (100 k out of more than 150 k examples) with a rigorous validation method that evaluates the robustness of the models against being trained on different training sets. Besides, one of our future works, will be applying our methodology and in particular LSTM-RNN modeling on many data sets from distant energy contexts and training more parameters of LSTM model via GA in a distributed manner. For instance, our approach could be applied for forecasting several decision variables such as the energy flow and the load in the context of urban vehicle networks [50]. Many alternatives of validation methods for our approach can be inspired by the extended discussion on validation in [51].

Finally and for the sake of reuse and duplication of experiments, the used code is available at: https://github.com/UAE-University/CIT_LSTM_TimeSeries.

7. Conclusions

Accurate forecasting of electricity consumption represents an essential part of energy management for sustainable systems. In this research, we built and optimized a LSTM-RNN-based univariate model for demand side load prediction, which forecasted accurately over both short-term (few days to 2 weeks) and medium term (few weeks to few months) time horizons. For comparison purposes, we implemented seven forecasting techniques representing a spectrum of commonly used machine learning approaches in the energy prediction domain. The best performing model on the studied consumption data set was used as our benchmark model. We explored wrapper and embedded techniques of feature selection namely recursive feature elimination and extra trees regressor to validate the importance of our model inputs. Hyperparameter tuning was used to further improve the

performance of ensemble-based machine learning model. These techniques unexpectedly eliminated the temperature and the quarter of year features and gave the most importance to historical time lags for forecasting energy consumption. This is due to the fact that temporal load parameters encapsulate information that is originally contained within the weather and weekday features. For time lagged features, a common problem is that of choosing the appropriate lags length which we solved using GA.

The validation of results using simple train test split, multiple train test splits and on various time horizons showed that the LSTM-RNN based forecasting method has lower forecast errors in the challenging short to medium term electric load forecasting problem compared to the best machine learning. We have modeled the electric load forecasting as a univariate time series problem and thus the approach can be generalized to other time series data. Our future works will include applying our methodology and in particular, LSTM-RNN modeling on many data sets from distant energy contexts. Another objective is to carry out an empirical study to identify the top techniques for each class of prediction circumstances.

Author Contributions: This paper is a collaborative work of the all the authors. Conceptualization, S.B. and A.F.; Methodology, S.B. and A.F.; Software, A.F.; Validation, S.B., A.F. and A.O.; Data Curation, A.F.; Writing-Original Draft Preparation, S.B., A.F., A.O. and M.A.S.; Writing-Review & Editing, S.B., A.F., A.O. and M.A.S.; Supervision, S.B.; Project Administration, S.B.; Funding Acquisition, S.B. and M.A.S.

Acknowledgments: The authors would like to acknowledge United Arab Emirates University for supporting the present wok by a UPAR grant under grant number G00001930 and providing essential facilities.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mocanu, E.; Nguyen, P.H.; Gibescu, M.; Kling, W.L. Deep learning for estimating building energy consumption. *Sustain. Energy Grids Netw.* **2016**, *6*, 91–99. [CrossRef]
2. Hyndman, R.J.; Shu, F. Density forecasting for long-term peak electricity demand. *IEEE Trans. Power Syst.* **2010**, *25*, 1142–1153. [CrossRef]
3. Chui, F.; Elkamel, A.; Surit, R.; Croiset, E.; Douglas, P.L. Long-term electricity demand forecasting for power system planning using economic, demographic and climatic variables. *Eur. J. Ind. Eng.* **2009**, *3*, 277–304. [CrossRef]
4. Hernandez, L.; Baladron, C.; Aguiar, J.M.; Carro, B.; Sanchez-Esguevillas, A.J.; Lloret, J.; Massana, J. A survey on electric power demand forecasting: Future trends in smart grids, microgrids and smart buildings. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1460–1495. [CrossRef]
5. Graves, A.; Jaitly, N. Towards End-To-End Speech Recognition with Recurrent Neural Networks. In Proceedings of the International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1764–1772.
6. Mao, J.; Xu, W.; Yang, Y.; Wang, J.; Huang, Z.; Yuille, A. Deep captioning with multimodal recurrent neural networks (m-RNN). *arXiv*, 2014.
7. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 3104–3112.
8. Hochreiter, S.; Schmidhuber, J. LSTM can solve hard long time lag problems. In Proceedings of the Advances in Neural Information Processing Systems, Denver, CO, USA, 2–5 December 1996; pp. 473–479.
9. Ribeiro, G.H.; Neto, P.S.D.M.; Cavalcanti, G.D.; Tsang, R. Lag selection for time series forecasting using particle swarm optimization. In Proceedings of the IEEE 2011 International Joint Conference on Neural Networks (IJCNN), San Jose, CA, USA, 31 July–5 August 2011; pp. 2437–2444.
10. Goldberg, D.E. *Genetic Algorithms in Search, Optimization, Machine Learning*; Addison Wesley: Reading, UK, 1989.
11. Wang, Z.; Srinivasan, R.S. A review of artificial intelligence based building energy use prediction: Contrasting the capabilities of single and ensemble prediction models. *Renew. Sustain. Energy Rev.* **2017**, *75*, 796–808. [CrossRef]
12. Liu, N.; Tang, Q.; Zhang, J.; Fan, W.; Liu, J. A hybrid forecasting model with parameter optimization for short-term load forecasting of micro-grids. *Appl. Energy* **2014**, *129*, 336–345. [CrossRef]

13. Weron, R. Electricity price forecasting: A review of the state-of-the-art with a look into the future. *Int. J. Forecast.* **2014**, *30*, 1030–1081. [CrossRef]
14. Ryu, S.; Noh, J.; Kim, H. Deep Neural Network Based Demand Side Short Term Load Forecasting. *Energies* **2017**, *10*, 3. [CrossRef]
15. Hagan, M.T.; Behr, S.M. The time series approach to short term load forecasting. *IEEE Trans. Power Syst.* **1987**, *2*, 785–791. [CrossRef]
16. Taylor, J.W. Short-term electricity demand forecasting using double seasonal exponential smoothing. *J. Oper. Res. Soc.* **2003**, *54*, 799–805. [CrossRef]
17. Taylor, J.W.; de Menezes, L.M.; McSharry, P.E. A comparison of univariate methods for forecasting electricity demand up to a day ahead. *Int. J. Forecast.* **2006**, *22*, 1–16. [CrossRef]
18. Park, D.C.; El-Sharkawi, M.; Marks, R.; Atlas, L.; Damborg, M. Electric load forecasting using an artificial neural network. *IEEE Trans. Power Syst.* **1991**, *6*, 442–449. [CrossRef]
19. Hernandez, L.; Baladrón, C.; Aguiar, J.M.; Carro, B.; Esguevillas, S.A.J.; Lloret, J. Short-term load forecasting for microgrids based on artificial neural networks. *Energies* **2013**, *6*, 1385–1408. [CrossRef]
20. Hippert, H.S.; Pedreira, C.E.; Souza, R.C. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Trans. Power Syst.* **2001**, *16*, 44–55. [CrossRef]
21. Box, G.E.; Jenkins, G.M.; Reinsel, G.C. *Time Series Analysis: Forecasting and Control*; John Wiley & Sons: Hoboken, NJ, USA, 2011; Volume 734.
22. Chen, J.-F.; Wang, W.-M.; Huang, C.M. Analysis of an adaptive time-series autoregressive moving-average (ARMA) model for short-term load forecasting. *Electr. Power Syst. Res.* **1995**, *34*, 187–196. [CrossRef]
23. Zhao, H.-X.; Magouls, F. A review on the prediction of building energy consumption. *Renew. Sustain. Energy Rev.* **2012**, *16*, 3586–3592. [CrossRef]
24. Foucquier, A.; Robert, S.; Suard, F.; Stephan, L.; Jay, A. State of the art in building modelling and energy performances prediction: A review. *Renew. Sustain. Energy Rev.* **2013**, *23*, 272–288. [CrossRef]
25. Cincotti, S.; Gallo, G.; Ponta, L.; Raberto, M. Modeling and forecasting of electricity spot-prices: Computational intelligence vs classical econometrics. *AI Commun.* **2014**, *27*, 301–314.
26. Amjady, N.; Keynia, F. Day ahead price forecasting of electricity markets by a mixed data model and hybrid forecast method. *Int. J. Electr. Power Energy Syst.* **2008**, *30*, 533–546. [CrossRef]
27. Bakirtzis, A.G.; Petridis, V.; Kiartzis, S.J.; Alexiadis, M.C.; Maissis, A.H. A neural network short term load forecasting model for the Greek power system. *IEEE Trans. Power Syst.* **1996**, *11*, 858–863. [CrossRef]
28. Papadakis, S.E.; Theocharis, J.B.; Kiartzis, S.J.; Bakirtzis, A.G. A novel approach to short-term load forecasting using fuzzy neural networks. *IEEE Trans. Power Syst.* **1998**, *13*, 480–492. [CrossRef]
29. Bashir, Z.; El-Hawary, M. Applying wavelets to short-term load forecasting using PSO-based neural networks. *IEEE Trans. Power Syst.* **2009**, *24*, 20–27. [CrossRef]
30. Kodogiannis, V.S.; Amina, M.; Petrounias, I. A clustering-based fuzzy wavelet neural network model for short-term load forecasting. *Int. J. Neural Syst.* **2013**, *23*, 1350024. [CrossRef] [PubMed]
31. Fan, S.; Chen, L. Short-term load forecasting based on an adaptive hybrid method. *IEEE Trans. Power Syst.* **2006**, *21*, 392–401. [CrossRef]
32. Bengio, Y.; Courville, A.; Vincent, P. Representation Learning: A Review and New Perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798–1828. [CrossRef] [PubMed]
33. Marino, D.L.; Amarasinghe, K.; Manic, M. Building energy load forecasting using Deep Neural Networks. In Proceedings of the IECON 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, Italy, 23–26 October 2016; pp. 7046–7051.
34. Rahman, A.; Srikumar, V.; Smith, A.D. Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. *Appl. Energy* **2018**, *212*, 372–385. [CrossRef]
35. Zheng, H.; Yuan, J.; Chen, L. Short-Term Load Forecasting Using EMD-LSTM Neural Networks with a Xgboost Algorithm for Feature Importance Evaluation. *Energies* **2017**, *10*, 1168. [CrossRef]
36. Roux, N.L.; Bengio, Y. Deep Belief Networks Are Compact Universal Approximators. *Neural Comput.* **2010**, *22*, 2192–2207. [CrossRef]
37. Colah.github.io. Understanding LSTM Networks—Colah's Blog. Available online: http://colah.github.io/posts/2015-08-Understanding-LSTMs (accessed on 5 April 2018).
38. Patterson, J.; Gibson, A. *Deep Learning. A Practitioner's Approach*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2017; pp. 150–158.

39. Wei, Y.; Zhang, X.; Shi, Y.; Xia, L.; Pan, S.; Wu, J.; Han, M.; Zhao, X. A review of data-driven approaches for prediction and classification of building energy consumption. *Renew. Sustain. Energy Rev.* **2018**, *82*, 1027–1047. [CrossRef]
40. Yildiz, B.; Bilbao, J.; Sproul, A. A review and analysis of regression and machine learning models on commercial building electricity load forecasting. *Renew. Sustain. Energy Rev.* **2017**, *73*, 1104–1122. [CrossRef]
41. RTE France. Bilans Électriques Nationaux. Available online: http://www.rte-france.com/fr/article/bilans-electriques-nationaux (accessed on 7 February 2018).
42. Dangeti, P. *Statistics for Machine Learning: Techniques for Exploring Supervised, Unsupervised, and Reinforcement Learning Models with Python and R*; Packt Publishing: Birmingham, UK, 2017.
43. Brooks, C. *Introductory Econometrics for Finance*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2008.
44. Hastie, T.J.; Tibshirani, R.J.; Friedman, J.H. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer: New York, NY, USA, 2009.
45. Huang, Y. Advances in Artificial Neural Networks—Methodological Development and Application. *Algorithms* **2009**, *2*, 973–1007. [CrossRef]
46. Scikit-learn.org. Parameter Estimation Using Grid Search with Cross-Validation—Scikit-Learn 0.19.1 Documentation. 2018. Available online: http://scikit-learn.org/stable/auto_examples/model_-selection/plotgrid_search_digits.html (accessed on 12 April 2018).
47. Lukoseviciute, K.; Ragulskis, M. Evolutionary algorithms for the selection of time lags for time series forecasting by fuzzy inference systems. *Neurocomputing* **2010**, *73*, 2077–2088. [CrossRef]
48. Sun, Z.L.; Huang, D.S.; Zheng, C.H.; Shang, L. Optimal selection of time lags for TDSEP based on genetic algorithm. *Neurocomputing* **2006**, *69*, 884–887. [CrossRef]
49. Scikit-learn.org. sklearn.model_selection.TimeSeriesSplit—Scikit-Learn 0.19.1 Documentation. 2018. Available online: http://scikitlearn.org/stable/modules/generated/sklearn.model_selection.Time-Series-Split.html (accessed on 18 April 2018).
50. Scellato, S.; Fortuna, L.; Frasca, M.; Gómez-Gardeñes, J.; Latora, V. Traffic optimization in transport networks based on local routing. *Eur. Phys. J. B* **2010**, *73*, 303–308. [CrossRef]
51. Bouktif, S. Improving Software Quality Prediction by Combining and Adapting Predictive Models. Ph.D. Thesis, Montreal University, Montreal, QC, Canada, 2005.